

Publicación práctica para usuarios de

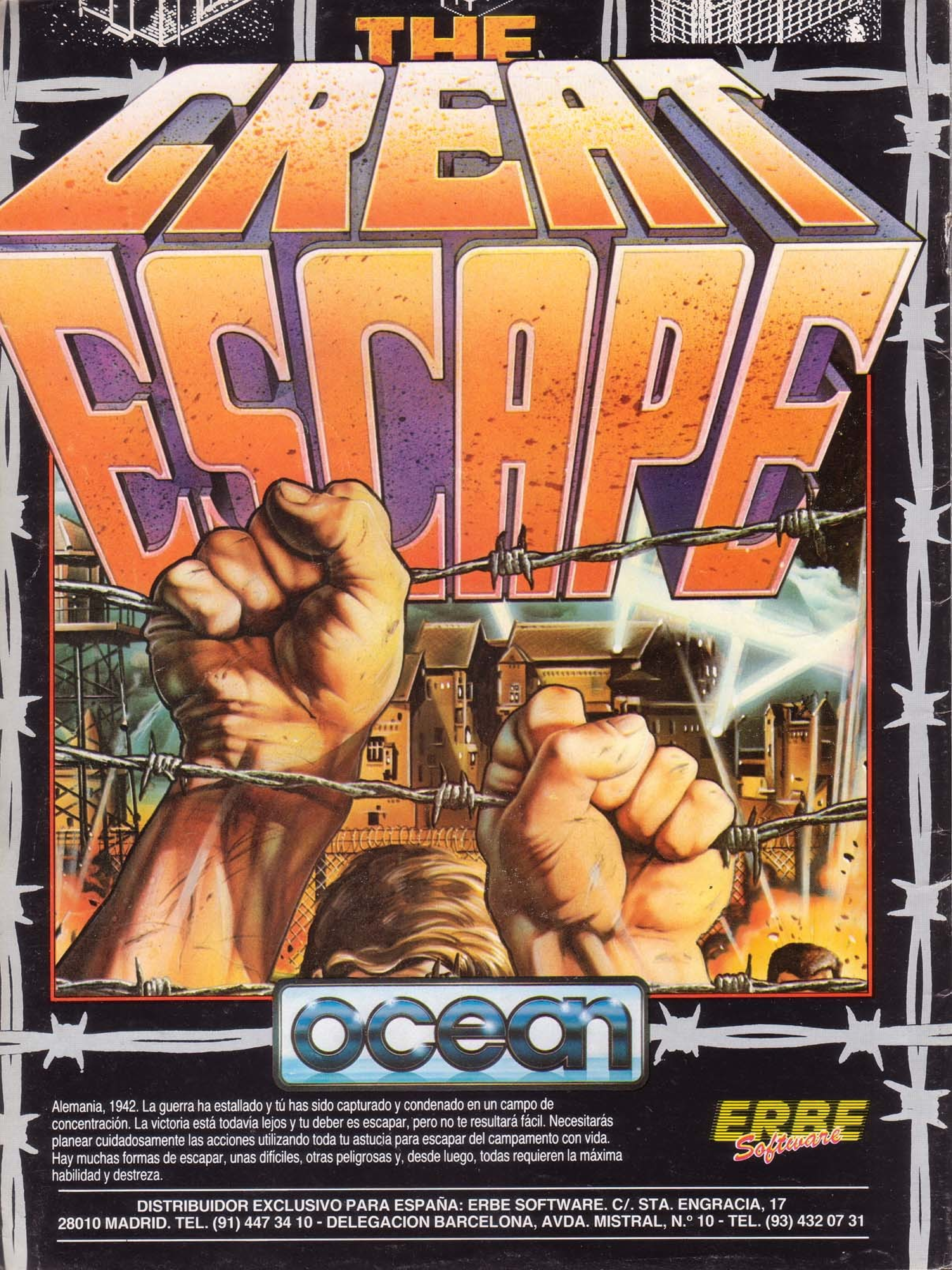
sinclair

Revista mensual 1986 Precio 350 Ptas Año 2 Número 14

CAJA DE HERRAMIENTAS EN CODIGO MAQUINA

PROCESADOR DE TEXTOS

EL MAPA DE LA ROM



THE
GREAT
ESCAPE
ocean
Alemania, 1942. La guerra ha estallado y tú has sido capturado y condenado en un campo de concentración. La victoria está todavía lejos y tu deber es escapar, pero no te resultará fácil. Necesitarás planear cuidadosamente las acciones utilizando toda tu astucia para escapar del campamento con vida. Hay muchas formas de escapar, unas difíciles, otras peligrosas y, desde luego, todas requieren la máxima habilidad y destreza.
ERBE Software
DISTRIBUIDOR EXCLUSIVO PARA ESPAÑA: ERBE SOFTWARE. C/. STA. ENGRACIA, 17
28010 MADRID. TEL. (91) 447 34 10 - DELEGACION BARCELONA, AVDA. MISTRAL, N.º 10 - TEL. (93) 432 07 31



AÑO 2 NUMERO 14

DIRECTOR:
Alejandro Diges
COORDINADOR EDITORIAL:
Francisco de Molina
DISEÑO GRAFICO:
Tomás López
COLABORADORES:
Antonio Taratiel, Luis R. Palencia, Francisco Tórtola, Benito Román, Esther de la Cal, Ernesto del Valle, Equipo Molisoft.

INPUT Sinclair es una publicación de PLANETA-DE AGOSTINI, S. A.

GERENTE DIVISION DE REVISTAS:
Angel Sabat

PUBLICIDAD: José Real-Grupo Jota
Madrid: c/ General Varela, 35
Teléf. 270 47 02/03
Barcelona: Avda. de Sarriá, 11-13, 1.º
Teléf. 250 23 99

FOTOMECANICA: Ochoa, S. A.

COMPOSICION: EFCA, S. A.

IMPRESION: Sirven Grafic
C/ Gran Vía, 754-756. 08013 Barcelona
Depósito legal: B-21954-1986

SUSCRIPCIONES: EDISA,
López de Hoyos, 141. 28002 Madrid
Teléf. (91) 415 97 12

REDACCION:
Paseo de la Castellana, 93, 14.ª
28046 Madrid. Teléf. 456 54 13

DISTRIBUIDORA
R.B.A. PROMOTORA DE EDICIONES, S. A.
Travesera de Gracia, 56. Edificio Odiseus.
08006 Barcelona.

El precio será el mismo para Canarias que para la Península y en él irá incluida la sobretasa aérea.

INPUT Sinclair es una publicación controlada por



INPUT sinclair

enter: Listado ensamblador y lo pone en memoria
1- Ensamblador y valor de las etiquetas
2- Solamente muestra en pantalla el ensamblado y el código pero no lo pone en memoria
3- igual 1 pero sin ponerlo en memoria
4- Lo ensambla y lo pone en memoria pero no lo muestra en pantalla
5- igual 4 pero lista valor etiquetas
6- Te indica si hay errores
7- igual 6 pero con lista etiquetas

SUMARIO

EDITORIAL **4**

ACTUALIDAD **5**

CODIGO MAQUINA
DEVPAK, ENSAMBLADOR DESENSAMBLADOR 6
RECETARIO DE CODIGO MAQUINA 14
RUTINAS DE LA ROM 28

APLICACIONES
REFERENCIAS CRUZADAS 18
CAJA DE HERRAMIENTAS 23
PROCESAMIENTO DE DATOS 40

PROGRAMACION
ESTRUCTURA TUS PROGRAMAS 44

REVISTA DE SOFTWARE **54**

EL ZOCO **64**

PROGRAMACION DE JUEGOS (COLECCIONABLE) **31**
LA APROXIMACION FINAL
SERPIENTES SUMADORAS

LA CRISIS DE LOS Ks

A estas alturas del año, cuando sólo nos resta por ver lo que pueda ofrecernos el **SIMO,** y ya se intuye la presión publicitaria de las campañas de Navidad, es el momento de hacer un repaso de lo que microinformáticamente ha sido el año 86. Lo iniciamos, ilusionadamente, con la llegada de los 128 K. Tanto **Commodore** como **Sinclair** parecían dispuestos a satisfacer la demanda de Ks de memoria que exigían los usuarios para sentir que sus máquinas no se quedaban atrás en el imparable avance de los micros por alcanzar las capacidades mitológicas de los grandes ordenadores. Sin embargo, en estos últimos meses, los fabricantes han decidido reconsiderar el camino que habrían de tomar sus nuevos productos.

Los que intuían una progresión geométrica de las capacidades de memoria que se proyectaba así hacia el infinito, se han encontrado con que **Commodore** ha diseñado una bonita carcasa para su clásico **C-64.** Y que **Sinclair-Amstrad,** aunque ha hecho bastante más por su **128** (le ha incorporado un cassette, un *chip* de sonido y alguna cosa más) no ha superado el listón de los 128 K alcanzado el invierno pasado ni en un solo bit. Si no hubiera sido por el *software* el año del Cometa, del **Amiga** y de la caída de los precios de los **PC** compatibles, habría sido un año gris para nuestros micros. Han sido los creadores de programas los que han dado el do de pecho y han puesto en evidencia las auténticas capacidades de estas magníficas máquinas. De su trabajo tenéis una amplia muestra en la revista de *software* de este número. Disfrutad con ellos de vuestro micro, y el año que viene ya veremos...

LOS MEJORES DE INPUT

Hemos pensado que es interesante disponer de un ***ranking*** que ponga en claro, mes a mes, cuáles son los programas preferidos de nuestros lectores. Para ello, es obligado preguntaros directamente y tener así el mejor termómetro para conocer vuestras preferencias. Podéis votar por cualquier programa aunque no haya sido comentado todavía en **INPUT**.

El resultado de las votaciones será publicado en cada número de **INPUT**.

Entre los votantes sortearemos 10 cintas de los títulos que pidáis en vuestros cupones.

Nota: No es preciso que cortéis la revista, una copia hecha a máquina o una simple fotocopia sirven.

Enviad vuestros votos a: **LOS MEJORES DE INPUT** P.º de la Castellana, 93. Planta 14. 28046 Madrid

ELIGE TUS PROGRAMAS

Primer título elegido ____________________

Segundo título elegido ____________________

Tercer título elegido ____________________

Programa que te gustaría conseguir ____________________

Qué ordenador tienes ____________________

Nombre ____________________

1.er Apellido ____________________

2.º Apellido ____________________

Fecha de nacimiento ____________________

Teléfono ____________________

Dirección ____________________

Localidad ____________________

Provincia ____________________

INPUT SINCLAIR N.º 14

Te estamos presentando el mejor SOFTWARE del año.



Una difícil misión tienes que acometer, antes de que destruyan toda la vida. El armamento más sofisticado se ha puesto a tu alcance.





PRODIGY nos introduce en el mundo "MEC" donde debemos conducir a "SOLO" el hombre sintético que cuida de "NEJO" y librarlo de los peligros más adversos, sin olvidarnos de WARDLOCK, el ser mecánico que quiere destruir toda vida orgánica.
Sus efectos sonoros y en 3 dimensiones lo hacen inmejorable y diferente.



The EIDOLON
LUCASFILM GAMES
ACTIVISION
HOME COMPUTER SOFTWARE

Descubre los secretos de EIDOLON a través de una misteriosa máquina del siglo XIX y sumérgete en un mundo subterráneo poblado de curiosas criaturas mágicas y feroces Dragones guardianes.



Disponibles para:

COMMODORE C
SPECTRUM S
AMSTRAD A

EN TIENDAS ESPECIALIZADAS Y GRANDES ALMACENES, O DIRECTAMENTE POR CORREO O TELEFONO A: **PROEIN, S.A.**
Distribuido en Cataluña por: DISCOVERY INFORMATIC C/. Arco Iris, 75 - BARCELONA - Tels. 256 49 08 / 09
Velázquez, 10 - 28001 Madrid - Tels. (91) 276 22 08/09

DEVPACK 3 ENSAMBLADOR DESENSAMBLADOR

Analizamos uno de los paquetes mas completos del mercado para trabajar en código de máquina con el ZX-Spectrum, **del que se puede sacar un gran provecho.**

Intentaremos también ofrecer unas explicaciones que consideramos más lógicas, concisas y manejables que las ofrecidas por el manual que acompaña al *cassette* con intención de complementarle. De hecho, nuestra sugerencia es que le consulte con frecuencia, al menos en principio.

La versión que revisamos es la última que ha salido al mercado español, aún cuando la casa inglesa **Hisoft** la publicó a finales de 1983 y ofrece sustanciales mejoras respecto a la anterior, como son el uso de **microdrive** y el manejo de **macros**.

Los programas ensamblador (**GENS3**) y desensamblador (**MONS3**) son totalmente independientes pero pueden utilizarse en forma combinada.

El conjunto es francamente flexible y potente, aunque en un principio, lógicamente, resulte un poco complejo de manejar. Esperamos que el trabajo que ofrecemos ayude a una mejor comprensión y utilización del paquete **DEVPACK 3**.

ENSAMBLADOR GENS3M21

El programa incluye un Editor de línea, un Ensamblador para **Z80**, y una serie de facilidades para manejar la carga y grabación con *cassette* o **microdrive**, así como la obtención de listados en impresora.

El programa en sí de la nueva versión ocupa 8866 bytes, frente a los 7K de la anterior.

FILOSOFIA DE FUNCIONAMIENTO Y ORGANIZACION DE LA MEMORIA

El **GENS3** es un ensamblador que maneja todos los nemónicos estándar del **Z-80** y permite, además, el uso de una serie de pseudo-nemónicos con los que manejar ensamblados condicionales, etiquetas, variables, etc.

Todo el proceso se realiza en dos pasadas. En la primera se detectan errores y se compila la «tabla de símbolos» (nombres de etiquetas y variables acompañadas de sus correspondientes valores). En la segunda pasada se efectúa la compilación, es decir, la generación del «programa objeto» o programa en código máquina.

El programa dispone de una serie de opciones y facilidades complementarias para poder listar tanto el «programa fuente» (texto) como el «programa objeto» (código) o la «tabla de símbolos».

Dentro de la flexibilidad de que hablábamos antes, existe la posibilidad de reubicar tanto el propio programa ensamblador como el código objeto (posiciones de memoria en las cuales va a residir durante su proceso normal de ejecución).

La distribución en memoria es la siguiente: primero se sitúa el propio **GENS3**, luego el *buffer* de macro (si se ha definido) a continuación el fichero de texto, después la tabla de símbolos y finalmente el código objeto (salvo que se haya definido otra posición mediante ORG).

Existen dos maneras de operar. En una de ellas el texto del programa fuente se genera mediante el teclado (con ayuda de Editor de línea), el cual se va almacenando en el fichero de texto. En la otra se «ensambla desde *cassette* o **microdrive**», es decir se utiliza un texto pregrabado (que se va pasando a la memoria de forma fraccionada). Con esta segunda forma de operar se consigue disminuir a voluntad el espacio requerido en memoria para el fichero de texto (*buffer Include*) y poder disponer de espacio para un programa objeto de mayor longitud, a cambio indudablemente de una mayor lentitud y complejidad en la operación.

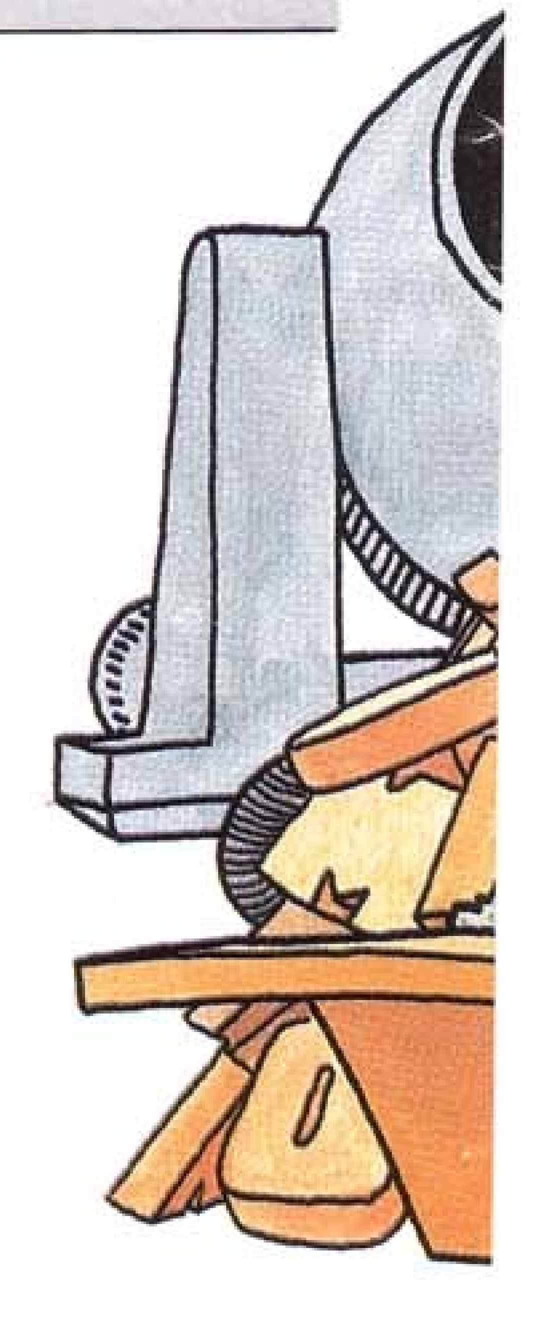
El ordenador calcula el espacio que prevé va a utilizar para el fichero de símbolos y normalmente es suficiente para la primera manera de operar, pero se puede alterar si se desea. En la segunda manera (ensamblado desde *cassette* o **microdrive**) siempre hay que definirlo ya que el ordenador no tiene forma de averiguar la longitud del programa ni la cantidad de símbolos que vamos a utilizar. En cualquier caso deberemos hallar un compromiso entre rapidez de ensamblado (mayor cuanto mayor sea el espacio reservado a texto) y espacio disponible para alojar el programa objeto.

En ambos casos deberá definirse el tamaño del *buffer* necesario para el manejo de macros (si es que se desea utilizarlos). Los detalles se verán más adelante.

PUESTA EN MARCHA

El programa ensamblador **GENS3** se carga con LOAD «» CODE x y se ejecuta inicialmente (y solo inicialmente) con RANDOMIZE USR x. Conviene utilizar el menor valor posible de x a fin de tener mayor margen de maniobra.

Las sucesivas veces que tratemos de ejecutar el programa podemos elegir entre un arranque «en frío» (RANDOMIZE USR x+56) o



una entrada «en caliente» (RANDOMIZE USR x+61). En el primer caso se limpia totalmente el fichero de texto, mientras que en el segundo se conserva el ya existente, situación que se nos presenta, por ejemplo, cuando nos hemos salido del programa para efectuar alguna corrección y deseamos continuar.

Tan pronto se ejecuta el programa, el control pasa automáticamente al EDITOR («<» en la pantalla).

EDITOR DE LINEAS

El **GENS3** incluye un editor de líneas muy completo y fácil de manejar. Con él podemos escribir y manejar el texto de nuestro programa fuente (línea a línea) con las ayudas correspondientes para insertar, renumerar, modificar, cancelar, grabar en/de *cassette* etc.

El texto que tecleamos para escribir cada línea de nuestro programa fuente (no hay que confundirlo con las líneas de pantalla correspondiente), no pasa directamente del teclado al fichero de texto como pudiera parecer sino que se almacena en un *buffer* intermedio de edición, donde puede ser modificado.

En el momento de pulsar ENTER es cuando el ordenador entiende que estamos conformes con el texto y comienza su paso al fichero, si bien durante este proceso se realiza una eliminación de huecos innecesarios a efectos de ahorro de espacio.

Toda línea de texto a procesar por el **GENS3** debe tener la estructura de la figura 1, aún cuando no siempre se requiere completar todos los campos:

	ETIQUETA	NEMONICO	OPERANDOS	COMENTARIO
Fig. 1	paso 1	LD	HL,var3	;comienzo proceso

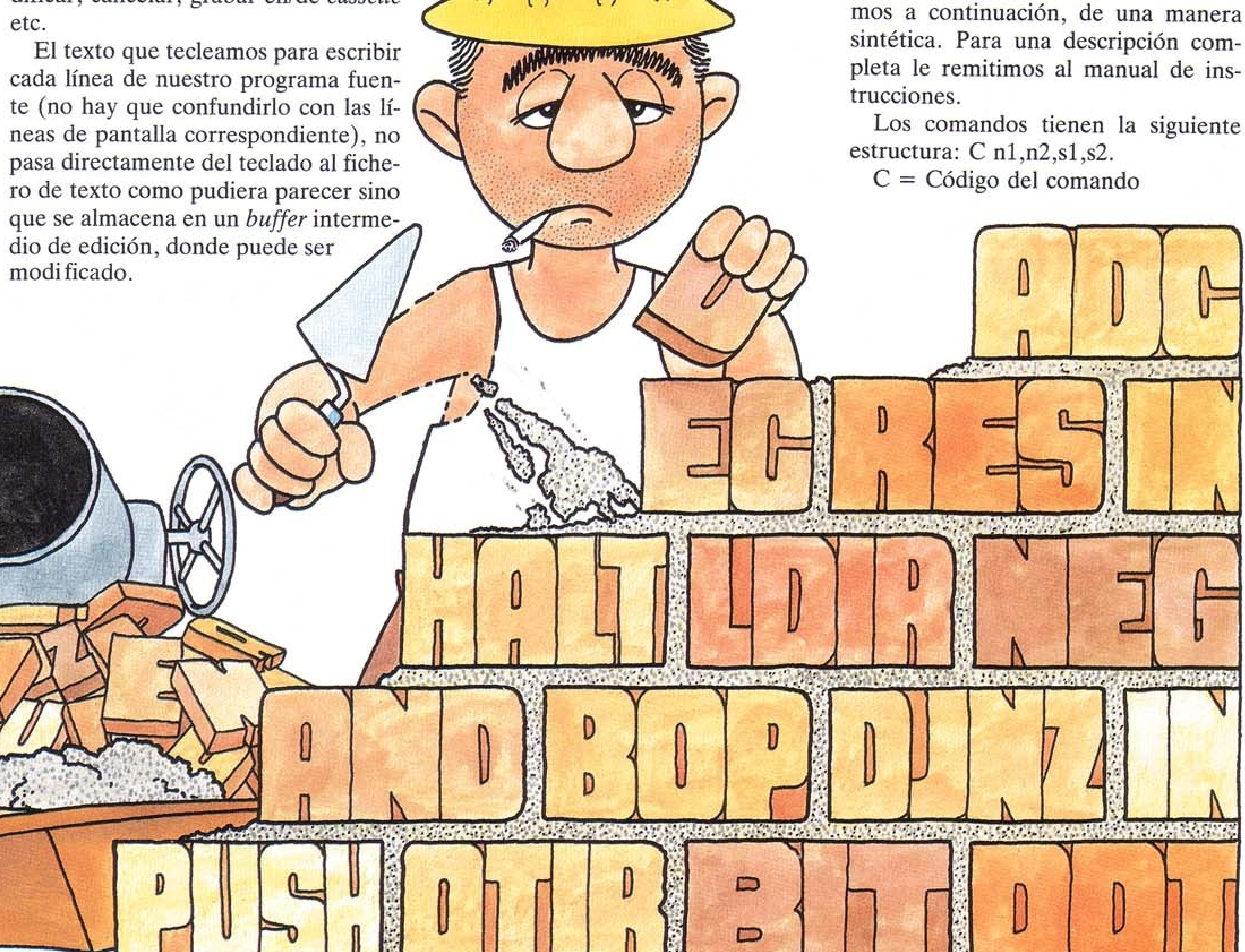
Además de la posibilidad de originar (editar) el texto del programa desde el teclado, también se puede manejar (reeditar) el existente en el fichero y hacer modificaciones (correcciones, cancelaciones o adiciones).

Cada línea del texto se puede introducir directamente tecleando primero el número correspondiente y a continuación la información de cada campo (CAPS SHIFT + 8 para introducir la tabulación requerida) y finalizando con ENTER . Una línea puede cancelarse con sólo escribir su número de orden seguido de ENTER

Normalmente será más cómodo hacer uso de los comandos del editor (inclusión, renumeración automática, inserciones, etc.) los cuales relacionamos a continuación, de una manera sintética. Para una descripción completa le remitimos al manual de instrucciones.

Los comandos tienen la siguiente estructura: C n1,n2,s1,s2.

C = Código del comando



n1 y n2 = parámetros numéricos (entre 1 y 32767)

s1 y s2 = cadenas de caracteres (20 caract. max.)

Dependiendo del comando se requerirán todos, sólo parte o ninguno de los parámetros. Inicialmente los parámetros tienen asignados el valor 10 para n1 y n2 y cadenas vacías para s1 y s2 (ver comando «X» para conocer en todo momento los valores asignados).

COMANDOS DEL ENSAMBLADOR

Son líneas del texto del programa fuente que no son decodificadas durante el ensamblado y que sólo tienen por objeto introducir modificaciones en el formato del listado.

Los comandos del ensamblador van siempre precedidos de «*» y se relacionan más abajo. Todos los comandos tienen acción sólo en la segunda pasada, excepto «*F» que la tiene en las dos.

***E** Envío de tres líneas en blanco

***Hs** El texto de la cadena «s» se imprime después de «*E»

***S** Produce una parada del listado (excepto después de *L−). Se continúa pulsando cualquier letra

***L−** Desactiva el listado e impresión

***L+** Activa el listado e impresión

***D+** Contador de posición de decimal

***D−** Contador de posición en hexadecimal

***C−** Listado abreviado (menos campos)

***C+** Listado completo (todos los campos)

***Fn** Permite ensamblar texto que se toma por bloques desde *cassette* o **microdrive**. «n» indica el nombre dado al fichero (max. 10 c). Si no se especifica «n», se carga el primer fichero. El texto en *cassette* o **microdrive** debe haberse cargado previamente con el comando «T» del editor, y el *Buffer Include* debe tener el mismo dimensionado que el original

***M+** Activa el listado de macros

***M−** Desactiva el listado de macros

PSEUDO-NEMONICOS DEL ENSAMBLADOR

Los pseudo-nemónicos que se indican a continuación son reconocidos como nemónicos por el ensamblador pero no originan programa objeto. Sólo ayudan a modificar éste.

A la hora de escribir el programa en ensamblador los pseudo-nemónicos tienen el mismo tratamiento que los nemónicos.

ORG expresión Cambia el Contador de Posición al valor de la expresión. Tener en cuenta opción 2 y 16. Esto permite colocar el código máquina en una posición que nos convenga para salvarlo posteriormente.

EQU expresión Asigna a la etiqueta que antecede el valor de la expresión. Ej. costo EQU 123.

DEFB exp,exp,.. Cambia el valor del byte (8 bits) indicado por el Contador de Posición al valor de la expresión (así para cada expresión). El C de P avanza 1.

DEFW exp,exp,.. Cambia el valor de los dos bytes indicados por el Contador de Posición al valor de la expresión (así para cada expresión). El C de P avanza 2.

DEFS expresión El C de P avanza lo indicado por expresión.

DEFM «cadena» Convierte cada caracter a su valor ASCII y los escribe en pos. de memoria sucesiva.

ENT expresión Fija la dirección de ejecución del programa objeto (situado a partir de ORG).

PSEUDO-NEMONICOS CONDICIONALES

Dan gran agilidad a la hora de escribir el programa fuente. Los pseudo-nemónicos IF, ELSE, END permiten seleccionar automáticamente aquellas secciones de texto que han de ensamblarse en cada circunstancia, ignorando las restantes.

IF expresión Si el resultado de la expresión es distinto de cero se efectúa el ensamblado. Si es cero se desactiva el ensamblado hasta encontrar «ELSE» o «END».

ELSE Hace de inversor. Si hasta ese momento el ensamblado estaba activo se cancela y viceversa.

END Activa el ensamblado después de haberse desactivado por un IF con expresión cero anterior.

Los pseudo-nemónicos condicionales no pueden anidarse.

MACROS

El macro puede considerarse como una subrutina o procedimiento genérico al que se puede acudir en cualquier momento, facilitando en cada caso los valores de los parámetros que se hayan definido al diseñar el macro.

Para conocer detalles sobre su manejo ver págs. 2a y 2b del manual.

OPCIONES

El autor del programa deja a nuestro criterio ejercer una serie de opciones. De esta manera nos adaptamos de una forma más flexible a nuestros requerimientos. Por ejemplo, durante la preparación de un programa necesitaremos ir probando diversos pasos para ver si se producen errores y en ese momento no deseamos perder tiempo con la generación del código objeto ni el listado del texto, para lo cual elegiremos Opción 6 (Opc. 2+4).

Existen 6 opciones independientes

Funcion	Comando	Comentarios	Salida
Insertar	I n,m	Inserción con renumeración automática a partir de 'n' con saltos de 'm'.	SHIFT+1
Renumerar	N n,m	Renumeración a partir de línea 'n' con intervalos de'm'.	
Listar	L n,m	Listado de línea 'n' a 'm' (por defecto 1,32767)	
	K n	Prepara para Listado en bloques de 'n'.	
Eliminar	D n,m	Eliminación de 'n' a 'm' (ambas inclusive)	
Copiar	M n,m	Pasa la línea 'n' a la 'm', sustituyendo a cualquier información existente. Conserva línea inicial	
Busqueda	Fn,m,f,s	Busca la serie 'f' para sustituirla por la 's' entre las líneas 'n' a 'm' (hace la búsqueda de la siguiente aparición o se procede a la sustitución dependiendo de que se ejecuten los comandos 'F' o 'S'.	
Edicion	E n	Selecciona línea 'n' para editar	
	Q	Deja todo como al principio	
	R	Recarga información original	
	L	Copia resto de la línea original	
	K	Borra el caracter del puntero	
	F	Localiza la siguiente serie fijada por F n,m,f,s	
	S	Sustituye la serie caracteres fijada por F n,m,f,s	
	I	Inserta caracteres en cursor	ENTER
	X	Restaura hasta el fin de línea y queda en modo I	
	C	Escribe sobre caracter del cursor	ENTER
Cassette	P n,m,s	Salva líneas 'n' a 'm' con nombre 's' (adecuado para añadir texto)	
	G,,s	Busca el fichero 's' y lo carga al final del texto actual	
	T n,m,s	Salva de 'n' a 'm' con nombre 's' (adecuado para Insertar texto pero no para añadir)	
Microdrive	H,,n:nomb.	Verifica el fichero de texto salvado en microdrive. 'n'=No. de microdrive,'nomb.'=nombre fichero	
Definicion de Buffer	C	Definición del tamaño de buffers Include y Macro.	
Ensamblado	A	Se ensambla el fichero de texto	
Ejecucion	R	Ejecución del programa objeto.	
Retorno	B	Retorno al sistema operativo del ZX-SPECTRUM.	
Camb.delim.	S,,d	Cambio de delimitador	
Val.defecto	V	Muestra valores por defecto de 'n1','n2','s1','s2'	
Impresora	W n,m	Listado de líneas 'n' a 'm' del fichero de texto (todo si no se especifican valores)	
Direccion de texto	X	Presentación decimal del comienzo y final del fichero de texto	

(pág. 3 del manual) que pueden combinarse, tal como se ha indicado.

ETIQUETAS

El uso de etiquetas (*label*) nos permite dirigirnos a una instrucción determinada del programa de una forma más cómoda y segura que si utilizamos direccionado absoluto. Si por ejemplo, queremos saltar en un cierto punto a la instrucción 345 y a esa instrucción la denominamos «calc1», podemos ordenar el salto a la línea 345 o a «calc1». Si posteriormente hemos tenido necesidad de incluir nuevas líneas, quizá nuestra instrucción ocupará ahora otro número y si no hemos tenido la precaución de modificar nuestra orden de salto seguiríamos yendo a la 345, con el correspondiente fallo del programa. Por el contrario las cosas seguirían marchando bien si hubieramos utilizado el direccionado a «calc1» ya que el programa se encargaría de actualizar la dirección.

Las etiquetas pueden utilizarse también como constantes en combinación con el pseudo-nemónico EQU.

Ver páginas 7 y 8 del manual.

CONTADOR DE POSICION (C de P)

El Contador de Posición puede inicializarse a cualquier valor mediante el pseudo-nemónico ORG y nos informa en todo momento de la posición en que nos encontramos dentro del programa. Sirve de referencia a la tabla de símbolos, nos permite hacer bifurcaciones, etc.

El símbolo «S» se utiliza como variable para conocer en cualquier momento el valor del Contador de Posición y utilizarlo como referencia.

TABLA DE SIMBOLOS

Se trata de una tabla de dos columnas donde en la primera se refleja el nombre de la etiqueta (max. 6 c.) y en la segunda el valor correspondiente a la definición de EQU o la posición del Contador de Posición.

La consulta a la tabla la realiza el ordenador de una forma muy rápida para conocer el valor posicional de una cierta etiqueta o el valor asignado a una variable cuyo nombre está en la tabla.

Ver páginas 8 y 9 del manual.

EXPRESIONES

En los operandos se pueden utilizar «términos» o «expresiones» compuestas por términos ligados por «operadores».

Los términos y operadores permitidos pueden verse en la parte correspondiente del manual (páginas 9, 10 y 11).

Hay que tener en cuenta que las operaciones se ejecutan en el orden en que se escriben, sin ningún tipo de prioridad o jerarquización.

Una expresión indicada entre pa-

LISTADO N.º 1

```
 10 *H"CAMBIO AUTOMATICO MINUSCULAS/MAYUSCULAS"
 20       ORG  60000
 30       LD   HL,(23635); CARGA EN HL DIR ORIGEN PROG.BASIC
 40       LD   DE,(23627); CARGA EN DE DIR VARIABLES
 50 SALTO INC  HL
 60       INC  HL
 70       INC  HL
 80       INC  HL
 90 CAMB  INC  HL
100 SIG-B AND  A
110       SBC  HL,DE
120       RET  NC
130       ADD  HL,DE
140       LD   A,(HL)
150       CP   13
160       JR   Z,SALTO
170       CP   14
180       INC  HL
190       JR   Z,SALTO
200       SUB  96          ; CAMBIAR POR 64 PARA MAYUS/MINUS
210       JR   C,SIG-B
220       SUB  26
230       JR   NC,SIG-B
240       ADD  A,90        ; CAMBIAR POR 122 PARA MAYUS/MINUS
250       DEC  HL
260       LD   (HL),A
270       JR   CAMB
280       RET
```

LISTADO N.º 2

```
                10 *H"CAMBIO AUTOMTICO MINUSCULAS/MAYUSCULAS"
EA60            20        ORG  60000
EA60 2A535C     30        LD   HL,(23635); CARGA EN HL DIR ORIGEN PROG.BASIC
EA63 ED5B4B5C   40        LD   DE,(23627); CARGA EN DE DIR VARIABLES
EA67 23         50 SALTO  INC  HL
EA68 23         60        INC  HL
EA69 23         70        INC  HL
EA6A 23         80        INC  HL
EA6B 23         90 CAMB   INC  HL
EA6C A7        100 SIG-B  AND  A
EA6D ED52      110        SBC  HL,DE
EA6F D0        120        RET  NC
EA70 19        130        ADD  HL,DE
EA71 7E        140        LD   A,(HL)
EA72 FE0D      150        CP   13
EA74 28F1      160        JR   Z,SALTO
EA76 FE0E      170        CP   14
EA78 23        180        INC  HL
EA79 28EC      190        JR   Z,SALTO
EA7B D660      200        SUB  96          ; CAMBIAR POR 64 PARA MAYUS/MINUS
EA7D 38ED      210        JR   C,SIG-B
EA7F D61A      220        SUB  26
EA81 30E9      230        JR   NC,SIG-B
EA83 C65A      240        ADD  A,90        ; CAMBIAR POR 122 PARA MAYUS/MINUS
EA85 2B        250        DEC  HL
EA86 77        260        LD   (HL),A
EA87 18E2      270        JR   CAMB
EA89 C9        280        RET
```

réntesis se refiere a una dirección de memoria.

ERRORES

Los errores que se cometen durante el manejo o ejecución de un programa están tipificados (ver págs. 2b y 26 del manual) y nos resultan de gran ayuda al darnos una pista de donde se encuentran nuestros problemas y de qué tipo son. Tan pronto se produce un error aparece en pantalla la información correspondiente.

PALABRAS RESERVADAS

Al asignar el nombre a las etiquetas tenemos gran libertad pero hay ciertos nombres que no podemos utilizar ya que el **GENS3** se los ha reservado para funciones específicas y daríamos lugar a errores de interpretación.

La lista de palabras reservadas puede verse en la pág. 27 del manual.

FUNCIONAMIENTO CON MICRODRIVE

Para aquellos que dispongan de **Interface 1** y **Microdrives** el manual dedica unas páginas de información con ejemplos ilustrativos (ver págs. 53 a 61 del manual).

CASO PRACTICO

Una vez que hemos visto y analizado cada uno de los aspectos más significativos vamos a trabajar con un caso práctico para ir decantando un poco las ideas.

Comenzaremos por cargar **GENS3** en la dirección 26000 con LOAD «» CODE 26000. Ejecutaremos RANDOMIZE USR 26000 y cuando aparezca «>» ya estamos en condiciones de trabajar.

Teclearemos inicialmente el programa en ensamblador cuyo listado (número 1) aparece en la página anterior.

Deberemos aprovechar la oportunidad para experimentar al máximo con todos y cada uno de los comandos que hemos visto. Cuando hayamos terminado con éxito esta operación procederemos a efectuar un listado en Impresora («W») o al menos en la pantalla («L»).

El programa con el que vamos a trabajar ahora es una rutina mediante la cual, cualquier programa en BASIC con texto en letras minúsculas/mayúsculas se puede modificar para que quede todo en mayúsculas (si modificáramos los valores de las instrucciones 200 y 240 podemos hacer lo contrario.

Una vez que dispongamos en código máquina (programa objeto) de esta rutina la podemos pasar a *cassette* para su uso posterior. La posición inicial en memoria se puede variar modificando ORG.

El programa (texto) lo tenemos almacenado en nuestro caso entre las direcciones 34866 y 35384 (ver comando «X») y podemos guardarlo en *cassette* con ayuda de los comandos «P» o «T». Con «T» si queremos ensamblar más tarde desde *cassette* y con «P» si más tarde deseamos añadirlo a otro pro-

grama o simplemente hacer un alto en nuestra sesión de trabajo para seguir en otro momento.

Vamos a proceder ahora al ensamblado («A»). Aparecerán los rótulos *Table size?* y *Options?*

Si el valor que elegimos para la tabla de símbolos es demasiado pequeño el ordenador nos contestará que no hay suficiente espacio (*No Table space*) y deberemos comenzar de nuevo. Si no indicamos ningún valor (ENTER), el ordenador elegirá uno que le irá bien (en nuestro caso 165 aunque sólo utilice 48). Si lo deseamos podemos elegir este valor y ahorrar algo de espacio en memoria.

En principio utilizaremos la Opción 1 (ó 9 si disponemos de impresora).

Si después de cada pasada no se ha producido ningún error aparecerán los letreros «*Pass 1 errors : 0*» y «*Pass 2 errors : 0*». En caso contrario deberemos verificar y corregir (*debug*).

Aparecerá un listado análogo al obtenido con «W» o «L» pero con dos campos más de información a la izquierda. (Ver listado 2.)

El primer campo (4 caracteres) indica las posiciones de memoria del código objeto en hexadecimal (añadiendo una instrucción *D+ aparecerá el valor decimal equivalente). El segundo campo muestra el valor almacenado en esta/s posición/es, también en hexadecimal. Ej. la instrucción No. 30 del ensamblador se traduce en los bytes 2A,53,5C (42, 83 y 92 decimal) y se aloja en las posiciones EA60, EA61 y EA62 (60000, 60001 y 60003 en decimal).

El programa objeto ocupa pues de la posición EA60 a la EA89 (60000 a 60041) es decir 42 bytes, y con estos datos podemos salvarlo en *cassette*. Estos 42 bytes los podríamos introducir en un programa cargador BASIC como DATA.

```
10 CLEAR 59999: FOR n=0 TO
   41
20 READ a: POKE 60000+n,a:
   NEXT n
30 DATA 42,83,92,237,91,75,
   92,35,35,35
40 DATA 35,35,167,237,82,208
   ,25,126,254,13,40
50 DATA 241,254,14,35,40,236
   ,214,96,56,237,214
60 DATA 26,48,233,198,90,43,
   119,24,226,201
70 DATA " # Input Sinclair
   # "
80 READ a$: RESTORE 70
90 PRINT a$
100 RANDOMIZE USR 60000
110 READ a$: PRINT a$
```

Como ejercicio no vendría mal al principio ensamblar «a mano» algunas instrucciones con ayuda del manual del **ZX Spectrum** y contrastar el resultado con las del **GENS**.

Si en cualquier momento posterior deseasemos cargar este mismo programa en otra dirección bastaría cargar nuevamente el fuente (comando G), modificar ORG a la dirección deseada, ensamblar y repetir nuevamente el proceso.

Por último observaremos que al final aparece la lista de etiquetas con sus direcciones correspondientes.

Conviene experimentar diversos valores de las posiciones **GENS** + 50, 51, 52, 53, 54/55 para adaptar el formato a nuestras necesidades.

Aunque ya se ha indicado conviene insistir en no olvidar efectuar un arranque en caliente (RANDOMIZE GENS+61) si no se desea perder el texto almacenado.

En cualquier caso la experiencia sólo se gana trabajando infinidad de casos y nuestra esperanza es la de haber contribuido a facilitar la labor.

En un próximo artículo nos dedicaremos a comentar el uso y funcionamiento del desensamblador **MONS3**.

GANADORES DE LOS MEJORES DE INPUT SINCLAIR

En el sorteo correspondiente al número 14 entre quienes escribisteis mandando vuestros votos a LOS MEJORES DE INPUT han resultado ganadores:

NOMBRE	LOCALIDAD	JUEGO ELEGIDO
Oscar Roncero Almazán	Madrid	Saboteur
Enrique Salavert Ferrer	Benetuser (Valencia)	Skyfox
J. Carlos Miranda Corraliza	Madrid	Dambusters
Rafela Sevilla López	El Altet (Alicante)	Commando
David Blanco Giro	Valladolid	The way of the tiger
Ramón Rodríguez Iglesia	Ponferrada (León)	Comando
Andrés Menéndez Fernández	Sevilla	Kung fu Master
Rubén Giménez Marcos	M. del Campo (Valladolid)	Kung fu Master
J. José Babot Lucena	S. Juan Despí (Barcelona)	Green Beret
Albert Arqués Perolada	Hostalric (Gerona)	Skyfox



MANEJO DE PANTALLA EN C/M (II)

Un buen programador ha de saber sacar provecho de cualquier programa para mejorar su técnica, incluso de los vulgarmente llamados «juegos» del Spectrum.

Sin duda uno de los puntos fuertes de este tipo de programas es el diseño gráfico. No te vamos a enseñar como dibujar mejor, pero si a fijarte un poco más. Si observas el proceso de carga de un juego tan famoso como el «Manic Miner», pronto repararás en la simple y vistosa pantalla de presentación. En ella parece que va alternando dos gráficos distintos simulando un cierto movimiento, a la vez que el programa continúa cargandose. El «secreto» de esta pantalla entra de lleno en el tema que nos ocupa.

El citado efecto se consigue manejando ciertos bytes de la memoria de atributos, zona que empieza en la posición 22528 y ocupa 768 bytes (uno por cada caracter que podemos escribir en pantalla). Cada uno de estos bytes contiene la información sobre la tinta, papel, brillo y *flash* de cada caracter representado, según la siguiente estructura: Los 3 primeros bits (el 0, el 1 y el 2) indican el color de la tinta, los tres siguientes el del papel, el bit 6 según esté a 0 ó a 1 indica el estado del brillo, y el séptimo bit determina la activación o desactivación del *flash* en cada caracter. Así por ejemplo, si deseas que los atributos del segundo caracter en pantalla sean tinta negra, papel amarillo, sin brillo pero con *flash*, POKEando directamente podrás conseguirlo si ejecutas:

POKE 22529,176 ya que 22529 es la dirección de memoria que contiene los atributos del segundo caracter, y 176 tiene la representación en binario. (Ver tabla 1.)

Una vez comprendido lo anterior estamos listos para entender la estuctura del programa. Lo único necesario es dibujar una serie de letras grandes a base de cuadros o «trocitos» de un byte de atributo. Si previamente hemos limpiado la pantalla, todo POKE que realicemos sobre la memoria de atributos quedará reflejado como un cuadrado monocolor de 8 por 8 *pixels*, y será bicolor si estamos utilizando el *flash*. Valiéndonos de esta última característica podemos conseguir la sensación de estar superponiendo dos planos distintos con sólo elegir los colores adecuados.

TABLA 1

```
                   :—>000 indica tinta negra
  10110000--:      —>110 papel amarillo(6 en binario)
  ↑         ↑      :—>0 (bit 6) brillo desactivado
  |         |      :—>1 (bit 7) FLASH 1
  bit 7     |
            bit 0
```

Además del efecto apuntado anteriormente, si cada plano que dibujamos tiene los caracteres desplazados una serie de posiciones con respecto al otro, con la aparición y posterior desaparición de cada uno de ellos se consigue crear esa ilusión de movimiento que buscábamos.

Situándonos en el ejemplo concreto que presentamos, vemos que sólo debemos POKEar en 102 de las 768 posiciones para conseguir el rótulo elegido. Tras limpiar la pantalla ponemos el borde, tinta y papel de color negro, que van a ser los atributos definitivos para los bytes en los que no realicemos un POKE posterior (recuerda que «INK x: PAPER x» usados como comandos generales es la manera más rápida de hacer un POKE colectivo para todos los atributos de pantalla. Sería equivalente también a introducir el valor en la variable del sistema ATTR-P).

Ahora antes de realizar los POKEs pertinentes debemos distinguir entre tres tipos de «cuadraditos» gráficos:

- La simulación del primer plano se consigue con bytes de la tinta elegida pero del color del fondo (negro), así por efecto del FLASH estarán alternando entre una tinta visible y un papel invisible (confundido con el fondo) por lo que el rótulo desaparecerá de la vista cuando cambie a papel negro.

- Si construimos ahora un segundo plano con bytes del papel que queramos pero con tinta negra, el efecto que conseguimos será el contrario al anterior, ya que ahora este plano sólo será visible cuando cambie a PAPER x, y se hará invisible cuando por efecto del FLASH pase a INK 0. Además hemos conseguido que este segundo plano se esté alternando con el primero, con lo que conseguimos que se vea uno o el otro, pero no los dos a la vez.

- La única dificultad posible ocurre cuando se superponen bytes de ambos planos. El proceso será el mismo, pero ahora esos cuadrados gráficos han de estar alternando entre la tinta de un plano y el papel del otro, por lo que nunca toman el color negro del fondo y son siempre visibles aunque en dos colores distintos.

Como ejemplo de todo lo anterior

nos podemos fijar que en la «I» que introduce el programa en el rótulo INPUT y la primera «I» del rótulo SINCLAIR se superponen a distinto nivel, por lo que al pasar de un gráfico a otro parece que dicha «I» está continuamente subiendo y bajando. La «I» de INPUT queda definida por bytes de tinta azul claro y papel negro, y la de SINCLAIR es una barra vertical compuesta por bytes de papel amarillo y tinta negra. Pero como hemos apuntado hay bytes de ambos que ocupan las mismas posiciones en pantalla, por lo que éstos quedarán definidos por papel amarillo y tinta azul clara. (Ver tabla 2.)

En este diagrama adjunto aparece el esquema de la superposición real en pantalla, donde se advierte que los tres «paquetes» o bytes centrales que forman la «I» toman el valor 245 correspondiente en binario a 11110101, es decir FLASH 1, BRIGHT 1, PAPER 6, INK 5.

Antes de introducirse de lleno en el programa y como última evidencia de su funcionamiento sugerimos probar:

```
10 BORDER 0: PAPER 0: INK 0:
   FLASH 0: BRIGHT 0: CLS
20 LET X=22820
30 FOR N=1 TO 7: READ A:
   POKE X,A: LET X=X+32:
   NEXT N
40 DATA 197,197,245,245,245,
   240,240
```

A continuación presentamos el programa en el que se ha tratado de trabajar con un sólo tercio de pantalla para conseguir reducir el número de DATAs por haber mayor cantidad de superposiciones. La única pega posible depende del rótulo elegido, el número de DATAs se multiplica por dos ya que las posiciones de memoria son totalmente aleatorias, no pudiendo construir un bucle llenarlas, por lo que debemos acompañar a cada valor a POKEar de su correspondiente dirección de memoria. Una vez tecleadas las DATAs y ejecutado el programa podemos grabar el resultado como CODE e introducirlo en nuestros programas como presentación, a modo del «Manic Miner».

```
10 BORDER 0: PAPER 0: FLASH
   0: BRIGHT 0: INK 0: CLS
20 FOR N=1 TO 102: READ P,H:
   POKE P,H: NEXT N: PAUSE 0
30 DATA 22817,208,22818,208,
   22819,208,22821,197,22823
   ,224,22827,224,22829,198,
   22830,198,22831,198,22832,
   198,22833,248,22839,199,
   22840,199,22841,231,22842,
   199,22843,199
40 DATA 22849,208,22853,197,
   22855,224,22856,224,22859,
   224,22861,198,22864,198,
   22865,248,22873,231
50 DATA 22881,208,22882,208,
   22883,208,22885,245,22887,
   226,22889,224,22891,226,
   22893,206,22894,206,22895,
   206,22896,198,22897,248,
   22898,196,22901,220,22902,
   216,22903,216,22905,231,
   22907,240,22908,240,22909,
   240,22910,240
60 DATA 22915,208,22917,245,
   22919,226,22920,194,22922,
   224,22923,226,22925,206,
   22929,248,22930,196,22933,
   220,22935,216,22937,231,
   22939,240,22942,240
70 DATA 22945,208,22946,208,
   22947,208,22949,245,22951,
   226,22953,194,22955,226,
   22957,206,22961,248,22962,
   252,22963,248,22965,220,
   22967,216,22969,231,22971,
   240,22972,240,22973,240,
   22974,240
80 DATA 22981,240, 22983,194,
   22986,194,22987,194,22989,
   200,22994,196,22997,220,
   22998,216,22999,216,23003,
   240,23005,240
90 DATA 23013,240,23015,194,
   23019,194,23021,200,23022,
   200,23023,200,23026,196,
   23027,196,23028,196,23029,
   220,23031,216,23035,240,
   23038,240
```

TABLA 2

197 ⟶	*				>>>>>>	*	⟶ 197
197 ⟶	*				>>>>>>	*	⟶ 197
197 ⟶	*	———	*	⟵ 240	>>>>>>	*	⟶ 245
197 ⟶	*	Superponen	*	⟵ 240	>>>>>>	*	⟶ 245
197 ⟶	*	———	*	⟵ 240	>>>>>>	*	⟶ 245
			*	⟵ 240	>>>>>>	*	⟶ 240
			*	⟵ 240	>>>>>>	*	⟶ 240
"I" de INPUT			"I" de SINCLAIR		Superposicion de ambas		

MANEJO DE ATRIBUTOS CON EL ENSAMBLADOR

Aunque en realidad el código máquina consiste en introducir en cada posición de memoria una serie de números, no siempre podemos realizar programas tan costosos como el anterior POKEando uno a uno los datos. Es mucho más limpio utilizar el ensamblador.

Si nuestro deseo es modificar toda la pantalla introduciendo un mismo color de papel, tinta, etc... lo podemos conseguir fácilmente utilizando uno de los bucles que acostumbramos a hacer, e introduciendo los datos en el formato de 8 bits explicado anteriormente. Pero con esto a lo sumo podemos obtener una pantalla aburrida monocolor o bicolor parpadeante.

Los conocimientos de la memoria de atributos nos permiten llevar a cabo otros procedimientos más refinados, como poner en FLASH o BRILLO la pantalla entera o parte de ella pero sin alterar para nada el archivo de pantalla, sólo el de atributos. Así nuestros escritos y dibujos seguirán a la vista pero resaltados o apagados según sea el caso.

RUTINA NUMERO 1

A continuación presentamos la rutina de activación del FLASH desde el c.m. Para ello se hace uso de la instrucción SET 7,(HL) que pone a 1 el bit 7 (el que controla el FLASH) de la posición de memoria apuntada por el registro doble HL, pero sin modificar los demás valores de papel, tinta, etc. (Ver rutina 1.)

El cargador BASIC de la rutina será:

```
10 CLEAR 59999: FOR N=60000
   TO 60015: READ A: POKE
   N,A: NEXT N
20 DATA 33,0,88,1,0,0,203,
   254,35,3,62,3,184,32,247,
   201
```

Una vez ejecutado el cargador podemos comprobar sobre un ejemplo su efecto:

```
10 FOR N=1 TO 704: PRINT
   PAPER (RND*7);INK (RND*7)
   ;"*": NEXT N: PAUSE 100:
   RANDOMIZE USR 60000
```

No sólo podemos modificar el FLASH, sino también mediante un sencillo cambio somos capaces de activar BRIGHT 1. Deberemos sustituir la línea 50 por:

50 SET 6,(HL) y en nuestra sentencia de DATAs el valor 254 por 246.

Pero no siempre queremos activar el FLASH o BRIGHT para toda la pantalla. Podemos separar la memoria de atributos en 3 tercios de 256 bytes y actuar sobre cada uno individualmente.

De la posición 22528 a 22783 → Primer tercio

De la posición 22784 a 23039 → Segundo tercio

De la posición 23040 a 23296 → Tercer tercio

Según lo visto anteriormente, para activar BRIGHT sólo en el tercio central deberemos cambiar las siguientes líneas del listado original:

```
30          LD HL,22784
50 BUCLE SET 6,(HL)
80          LD A,1
```

y alterando los tres valores indicados la sentecia DATA quedará:

```
20 DATA 33,0,89,1,0,0,203,
   246,35,3,62,1,184,32,247
   ,201
```

Se puede conseguir acortar aún más el número de bytes activados por la rutina con sólo modificar el contador, llegando incluso a poder destacar sólo una frase o un dibujo importante.

RUTINA NUMERO 2

Existe una forma de activar a la vez el FLASH y BRIGHT sin duplicar la rutina, y es utilizando las funciones lógicas AND, OR, XOR. Podemos recordar sus operaciones observando el cuadro de la página siguiente.

RUTINA 1

```
 10        ORG 60000    ;Se coloca la rutina a partir de la
 20        ENT 60000    ;posicion 60000
 30        LD  HL,22528;Comienzo de la memoria de atributos
 40        LD  BC,0     ;Posiciona en 0 el contador
 50 BUCLE SET 7,(HL)   ;Pone a 1 el bit 7 (FLASH 1)
 60        INC HL       ;Apunta asi a la siguiente posicion
 70        INC BC       ;Incrementa el contador
 80        LD  A,3      ;Si se da el caso que B=3 al ser el byte
 90        CP  B        ;alto de BC, el proceso se habra repetido
100        JR  NZ,BUCLE;256*3=768 veces, y se activara el flag Z
110        RET          ;devolviendo el control al BASIC.
```

RUTINA 2

```
 10        ORG 60000    ;La rutina es semejante a la anterior
 20        ENT 60000    ;pero con la diferencia esencial de que
 30        LD  HL,22784 ;ahora el valor contenido en la posicion
 40        LD  BC,0     ;indicada por HL es pasado al registro
 50 BUCLE LD  A,(HL)   ;simple A, donde siempre es necesario
 60        OR  %11000000;introducirlo antes de llevar a cabo la
 70        LD  (HL),A   ;funcion logica. El resultado es tambien
 80        INC HL       ;almacenado en el acumulador. La
 90        INC BC       ;instruccion 70 ejecuta la tarea de
100        LD  A,3      ;devolver el valor a la posicion de
110        CP  B        ;memoria de donde fue extraido, pero ya
120        JR  NZ,BUCLE ;modificado. Es la accion equivalente al
130        RET          ;"pokear" del BASIC
```

AND

	0	1
0	0	0
1	0	1

OR

	0	1
0	0	1
1	1	1

XOR

	0	1
0	0	1
1	1	0

La solución que nos interesa es la utilización de OR. Viendo la forma en que actúa podemos predecir que sea cual sea el número que tratemos, si hacemos un OR con 11000000 los bits 6 y 7 que son los que nos interesan quedarán activados, y los restantes inalterados. (Ver rutina 2.)

El cargador BASIC quedará:

```
10 CLEAR 59999: FOR N=60000
   TO 60017: READ A: POKE N,
   A: NEXT N
20 DATA 33,0,89,1,0,0,126,
   246,192,119,35,3,62,1,184
   ,32,245,201
```




COMO HACER REFERENCIAS CRUZADAS

- EDICION DE UN PROGRAMA EN BASIC
- SEGUIMIENTO DE ERRORES
- BUSQUEDA DE CADENAS
- BUSQUEDA CON SUSTITUCION
- BUSCA UNA PALABRA RESERVADA



Con esta práctica utilidad en código máquina podrás localizar en pocos segundos cualquier cadena de caracteres o palabra reservada dentro de un programa en BASIC, ahorrándote gran cantidad de tiempo y esfuerzo. También podrás hacer la sustitución de una cadena por otra.

El programa de referencias cruzadas que presentamos es lo que se llama un programa de utilidades o simplemente una «utilidad». Esto significa que no hace nada él solo, sino que se utiliza como una herramienta para ayudarte a desarrollar otros programas. Por esta razón está escrito en lenguaje máquina, a fin de que pueda ser cargado en el ordenador coexistiendo con un programa en BASIC. El programa en lenguaje máquina se queda en la memoria del ordenador, esperando a que se le llame cada vez que haga falta.

El programa se utiliza para realizar búsquedas de textos, con lo que te podrás librar de tener que seguir penosamente la pista a alguna determinada sección de un programa. Si le dices lo que estás buscando —una cadena de caracteres, una palabra reservada, un nombre de variable o lo que sea— el programa se encargará de buscarlo a lo largo del programa en BASIC y presentará todas las líneas en las que aparezca. Aparte de esto, también cuenta con una facilidad extra de «sustitución», que te permitirá remplazar una cadena por otra en todo el programa.

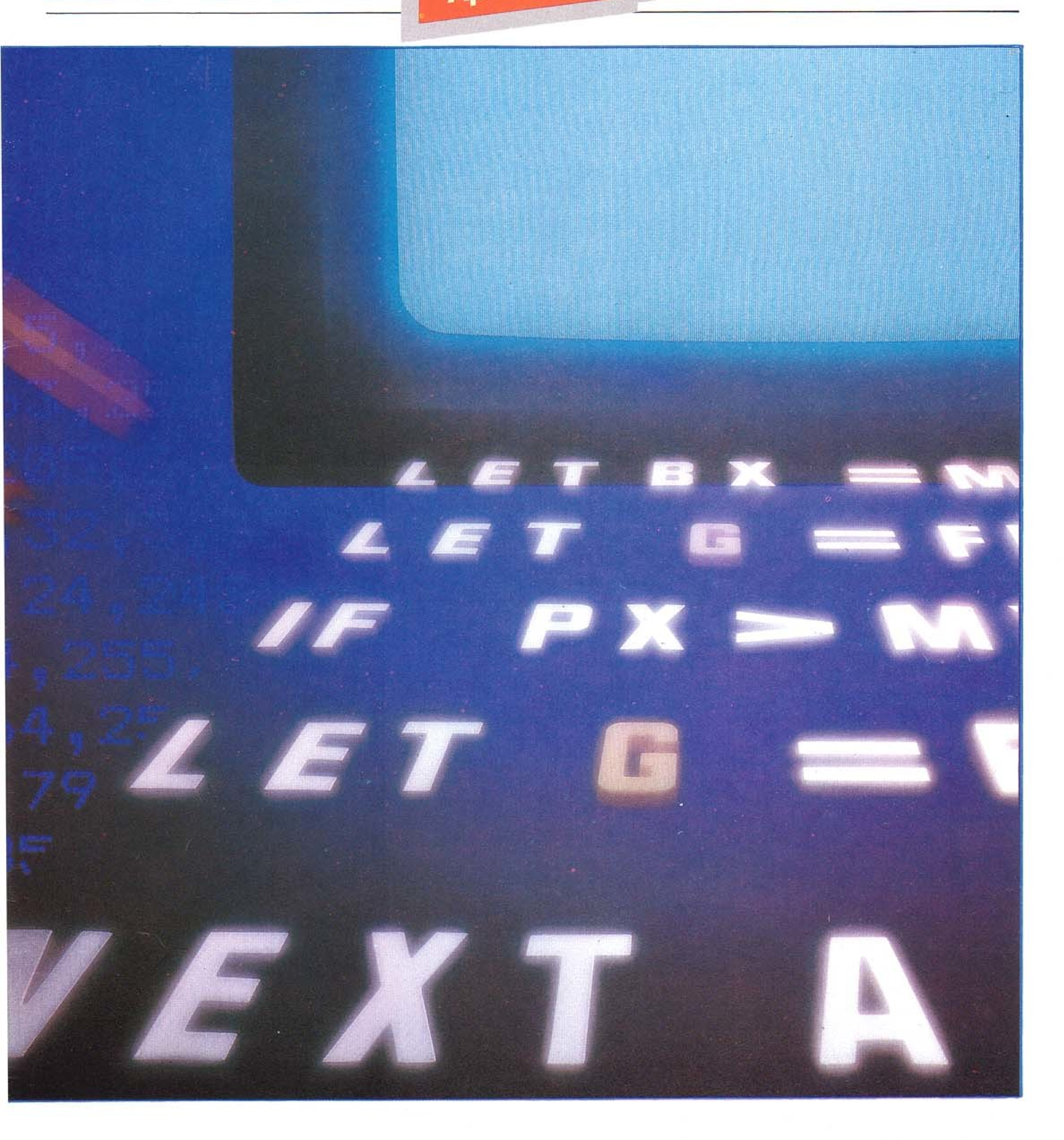
El programa te resultará de la mayor utilidad cuando estés desarrollando o escribiendo un programa en BASIC. Por ejemplo, si llegas a la conclusión de que tienes que cambiar algunas de las sentencias PRINT o INPUT, el programa te listará todas las líneas en que aparezcan dichas sentencias. O si te das cuenta de que el nombre de una de las variables de tu programa entra en conflicto con otras rutinas, puedes usar la opción de búsqueda con sustitución, cambiando el nombre por otro, sin tener que editar ni una línea.

También te resultará muy útil tener cargado en el ordenador el programa de las referencias cruzadas cuando estés intentando adaptar o comprender un programa escrito por otra persona. Por ejemplo, puede ser un ejercicio muy fructífero imprimir todas las líneas en que aparecen las palabras GOTO o GOSUB, ya que a partir de aquí rápidamente podrás trazar la secuencia de sucesos.

También te ayudará a localizar y perseguir los errores de programación que pueda contener tu programa. Por ejemplo, puede ser que descubras que una determinada variable tome valores incorrectos en alguna parte. El procedimiento usual de trabajo en estos casos es buscar línea por línea por todo el programa para localizar todos los sitios en que aparece dicha variable; es más que probable que se te escapen unas cuantas. Sin embargo el programa de referencias cruzadas hará este trabajo por tí con toda precisión en cuestión de segundos.

El programa de referencias cruzadas se complementa con el de Herramientas publicado en este mismo número. No obstante, la parte de referencias cruzadas es completamente independiente y si quieres puedes utilizarla separadamente. El programa contiene un pequeño subprograma cargador en BASIC seguido por el código máquina en sentencias DATA. Teclea el programa en BASIC, sálvalo y ejecútalo (con RUN); suponiendo que todo está bien hecho, el programa también salvará el lenguaje máquina resultante.

Esta versión en código máquina es lo que necesitas para utilizar el programa. Para cargarlo, teclea:




```
CLEAR 64559
LOAD "CREF" CODE
```

Puedes cargarlo antes o después del programa en BASIC sobre el que quieras actuar, en ambos casos el procedimiento es el mismo. Una vez cargado el programa, puedes llamar a las diferentes opciones utilizando las llamadas de RANDOMIZE USR tal como se describe en los siguientes párrafos.

Para buscar una cadena de caracteres, teclea RANDOMIZE USR 64634. Te aparecerá el mensaje «Introducir cadena a buscar». Cuando hayas tecleado la cadena de caracteres que quieres buscar y pulsado ENTER , el programa te presentará todas las líneas en que aparezca dicha cadena.

Para buscar una palabra reservada del BASIC, teclea RANDOMIZE USR 64911. Nuevamente te pedirá el programa la cadena de caracteres que hay que buscar, en este caso la palabra reservada, y presentará todas las líneas que contengan dicha palabra.

Para sustituir el nombre de una variable por otro nombre, teclea RANDOMIZE USR 64796. En esta ocasión tienes que teclear el nombre de la cadena buscada y el nuevo nombre que le sustituirá. Ambas cadenas pueden tener hasta un máximo de 20 caracteres y no es preciso que sus longitudes sean iguales. La sustitución especificada se hace automáticamente en todo el programa, pero esta vez no se presentan las líneas en la pantalla.

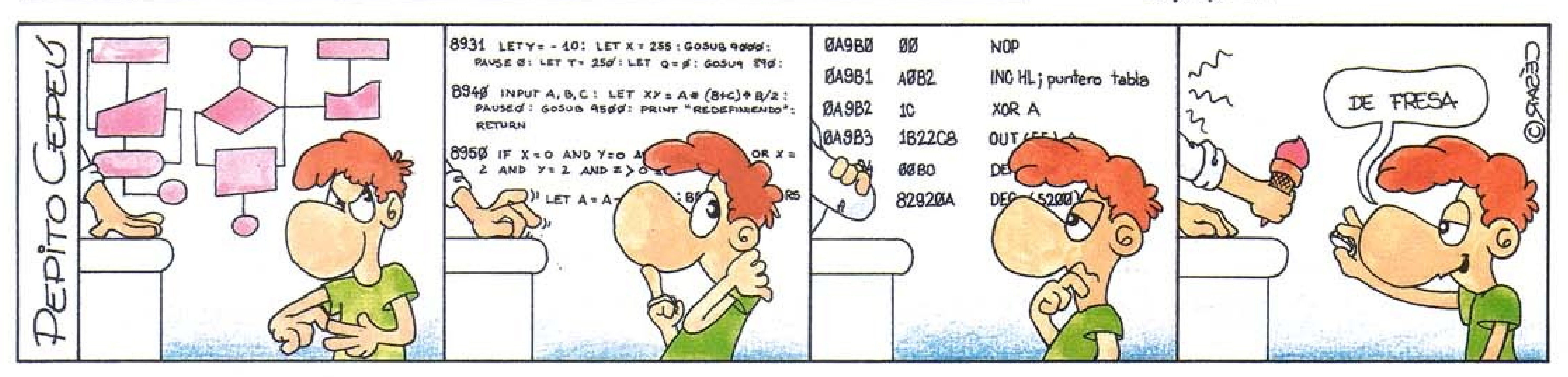
La última opción te permite ver todas las líneas en que se ha definido una función (utilizando DEFFN) o se ha realizado una llamada a función. Para seleccionar esta opción, pulsa RANDOMIZE USR 64713.

A continuación tienes el listado del programa. Si tienes un **Microdrive**, cambia la palabra *«cassette»* en la línea 30 por «cartucho» y modifica la línea 87 escribiéndola así: 87 SAVE*«M»;1;«CREF»CODE 64560, 832.

```
10 CLEAR 64559: BORDER 0:
   INK 7: PAPER 0: CLS
20 PRINT  INVERSE 1;AT 0,6;
   " REFERENCIAS CRUZADAS "
30 PRINT '''''" POKEANDO EN
   CODIGO MAQUINA.[5*
   ESPACIO]PREPARA UN
   CASETTE PARA SALVAR"
40 LET L=90: RESTORE L: FOR
   N=64560 TO 65343 STEP 16
50 LET T=0: FOR D=0 TO 15:
   READ A: POKE N+D,A: LET
   T=T+A: NEXT D
60 READ A: IF A<>T THEN
   PRINT  FLASH 1;" ERROR EN
   LINEA";L;"!": PRINT
   "VALOR ESPERADO";A;"VALOR
   ACTUAL ";T: STOP
70 LET L=L+10: NEXT N
80 PRINT AT 18,0;"PULSA UNA
   TECLA PARA SALVARLO"
85 LET A$=INKEY$: IF A$=""
   THEN  GO TO 85
87 SAVE "CREF"CODE 64560,
   832
88 STOP
90 DATA 203,39,95,22,0,221,
   33,233,253,221,25,221,
   110,0,221,102,1999
100 DATA 1,205,69,252,201,62
    ,254,229,205,1,22,225,
    126,35,254,255,2396
110 DATA 40,4,215,195,76,252
    ,201,207,9,209,225,205,
    229,25,201,205,2498
120 DATA 142,2,14,0,32,249,
    205,30,3,48,244,21,95,
    205,51,3,1344
130 DATA 254,0,201,46,25,118
    ,45,32,252,201,62,13,205
    ,48,252,33,1787
140 DATA 63,255,17,85,255,
    205,3,253,42,83,92,34,59
```

```
    ,255,126,254,2081
150 DATA 64,208,17,4,0,25,
    126,254,13,32,3,35,24,
    237,205,179,1426
160 DATA 252,56,3,35,24,240,
    229,42,59,255,205,85,24,
    62,13,215,1799
170 DATA 225,24,240,229,17,
    85,255,58,63,255,71,26,
    19,190,35,32,1824
180 DATA 5,16,248,225,55,201
    ,225,191,201,62,254,205,
    1,22,42,83,2036
190 DATA 92,34,59,255,126,
    254,64,208,17,4,0,25,126
    ,254,13,32,1563
200 DATA 3,35,24,237,254,168
    ,40,7,254,206,40,3,35,24
    ,237,229,1796
210 DATA 42,59,255,205,85,24
    ,62,13,215,225,62,13,1,0
    ,0,237,1498
220 DATA 177,24,206,175,119,
    213,229,205,115,252,205,
    95,252,245,215,241,2968
230 DATA 225,209,254,13,40,5
    ,18,19,52,24,234,201,62,
    13,205,48,1622
240 DATA 252,17,85,255,33,63
    ,255,205,3,253,62,18,205
    ,48,252,33,2039
250 DATA 64,255,17,65,255,
    205,3,253,42,83,92,126,
    254,64,208,17,2003
260 DATA 4,0,25,126,254,13,
    32,3,35,24,240,254,34,32
    ,9,35,1120
270 DATA 1,0,0,237,177,126,
    24,243,205,179,252,56,3,
    35,24,227,1789
280 DATA 58,63,255,79,58,64,
    255,145,56,25,40,8,79,6,
    0,229,1420
290 DATA 205,85,22,225,58,64
    ,255,17,65,255,79,6,0,
    235,237,176,1984
300 DATA 235,24,192,237,68,
    79,6,0,229,205,232,25,
    225,24,229,62,2072
310 DATA 13,205,48,252,17,85
    ,255,33,63,255,205,3,253
    ,33,150,0,1870
320 DATA 27,235,203,254,235,
    1,165,90,197,205,179,252
    ,56,13,35,203,2350
330 DATA 126,35,40,251,193,
    12,16,240,195,87,252,193
    ,42,83,92,34,1891
340 DATA 59,255,126,254,64,
    208,35,35,35,35,126,35,
    254,13,40,239,1813
350 DATA 185,32,247,197,62,
    13,1,0,0,237,177,229,42,
    59,255,205,1941
360 DATA 85,24,62,13,215,225
    ,193,24,214,15,254,70,
    254,79,254,86,2067
370 DATA 254,98,254,110,254,
    27,254,35,254,48,254,64,
    254,124,254,143,2681
380 DATA 254,160,254,184,254
    ,206,254,229,254,236,254
    ,0,255,10,255,66,3125
390 DATA 89,84,69,83,32,70,
    82,69,69,61,255,80,82,79
```

```
       ,71,82,1357
400 DATA 65,77,255,78,85,77,
    66,69,82,32,65,82,82,65,
    89,255,1524
410 DATA 67,72,65,82,65,67,
    84,69,82,32,65,82,82,65,
    89,255,1323
420 DATA 66,89,84,69,83,255,
    80,82,79,71,82,65,77,61,
    255,32,1530
430 DATA 66,89,84,69,83,255,
    70,73,76,69,32,84,89,80,
    69,61.1349
440 DATA 32,255,70,73,76,69,
    32,78,65,77,69,61,32,
    255,70,73,1387
450 DATA 76,69,32,76,69,78,
    71,84,72,61,32,255,69,78
    ,84,69,1275
460 DATA 82,32,83,84,65,82,
    84,32,76,73,78,69,32,58
    ,255,69,1254
470 DATA 78,84,69,82,32,69,
    78,68,32,76,73,78,69,32
    ,58,255,1233
480 DATA 69,78,84,69,82,32,
    76,73,78,69,32,73,78,67
    ,82,69,1111
490 DATA 77,69,78,84,83,32,
    58,255,69,78,84,69,82,
    32,84,65,1299
500 DATA 82,71,69,84,32,83,
    84,82,73,78,71,58,32,
    255,69,78,1301
510 DATA 84,69,82,32,68,69,
    67,73,77,65,76,32,78,85
    ,77,66,1100
520 DATA 69,82,32,58,255,72
    ,69,88,32,61,32,255,69,
    78,84,69,1405
530 DATA 82,32,72,69,88,39,
    32,78,85,77,66,69,82,58
    ,32,255,1216
540 DATA 68,69,67,73,77,65,
    76,61,32,255,69,78,84,69
    ,82,32,1257
550 DATA 78,69,87,32,83,84,
    82,73,78,71,58,255,0,0,
    0,0,1050
560 DATA 9,23,220,10,254,21
    ,206,11,254,80,3,23,220
    ,10,215,24,1583
570 DATA 177,1,0,10,0,100,0
    ,232,3,16,39,48,48,49,
    48,0,771
```



LA CAJA DE HERRAMIENTAS

- INCORPORACION DE NUEVOS COMANDOS
- RUTINA DE RENUMERACION
- NUMERACION AUTOMATICA DE LINEAS
- SUPRESION DE BLOQUES
- OTROS COMANDOS

Haz que tu vida resulte un poco más fácil con este nuevo juego de herramientas de programación. Esta utilidad en código máquina te ofrece muchas rutinas que te ayudarán a configurar tus programas.

Aunque todos los ordenadores utilizan en principio el mismo lenguaje BASIC, te habrás dado cuenta de las variantes que existen de unos dialectos a otros. De hecho, es muy raro encontrar un programa, por corto que sea, que corra en más de un ordenador. A veces no hace falta modificar más que la forma en que se utilizan los comandos, pero muchas veces te encontrarás que muchos comandos implementados en uno de los ordenadores no existen en absoluto en el otro. Muchos de los comandos no disponibles son precisamente los que, sin ser absolutamente esenciales para la programación, hacen que la vida resulte más fácil para el programador, al incluir facilidades como la renumeración, la autonumeración de líneas, o la conversión de hexadecimal a decimal, etc.

El programa de caja de herramientas para el **Spectrum** que vamos a ver a continuación incorpora estos comandos y algunos otros, facilitándote grandemente el uso y la programación de tu ordenador. Hay otros ordenadores que no necesitan estas utilidades, ya que disponen de ellas en su BASIC estándar.

El programa está escrito en código máquina, por lo que puede permanecer en tu ordenador al mismo tiempo que un programa en BASIC. Las instrucciones necesarias para cargar, almacenar y utilizar el programa se dan en las siguientes secciones.

En este juego de utilidades de programación te ofrecemos ocho nuevas rutinas: renumeración, supresión de bloques, bytes libres, longitud de programa, autonumeración de líneas, catálogo de cintas y conversiones de hexadecimal a decimal o viceversa. Todas estas funciones se llaman con ayuda de RANDOMIZE USR, seguido de un número, tal como se describe a continuación. Después de llamar a una rutina, el programa te enviará un mensaje para que introduzcas las entradas pertinentes, tal como son las líneas que han de ser destruídas, el número a que hay que convertir, o lo que sea.

Las utilidades emplean varias de las rutinas del programa **Referencias cruzadas,** publicado en este número, por lo que tienes que utilizar estos dos programas juntos. Todas las instrucciones para hacer esto están contenidas dentro del presente programa. En consecuencia, no tienes más que teclearlo, ejecutarlo y atender a las indicaciones que te vayan apareciendo en pantalla. El programa te informará de si has cometido algún error al introducir los DATA. Tienes que corregir los posibles errores antes de almacenar el código máquina. El programa almacenará el conjunto de utilidades y el programa de referencias cruzadas bajo el nombre «HERRAMIENTAS» CODE. Para cargarlo al principio de una sesión de programación, utiliza:






```
CLEAR 63488
LOAD "HERRAMIENTAS" CODE
```

La rutina RENUMBER se llama por medio de RANDOMIZE USR 63489. Recibirás un mensaje pidiéndote que introduzcas los incrementos de líneas, al que puedes responder con cualquier número entre 1 y 255.

Para encontrar la longitud de tu programa, teclea:

```
RANDOMIZE USR 63889
```

Si lo que quieres saber es el número de bytes libres, teclea lo siguiente:

```
RANDOMIZE USR 63860
```

La numeración automática de líneas se llama por medio de RANDOMIZE USR 64154. Te aparecerá un mensaje pidiéndote la línea de partida (1 a 9900) y los incrementos de líneas (1 a 9900). Para cancelar esta función AUTO, teclea dos ceros cuando aparezca el número de línea. La rutina se detendrá, enviándote un mensaje de «no tiene sentido en BASIC» que puedes ignorar.

Para la destrucción de bloques tienes que llamar a RANDOMIZE USR 64000. Los mensajes que aparezcan te pedirán que introduzcas los números

de líneas inicial y final de la sección que quieras suprimir.

Para tener el catálogo de cintas, llama a RANDOMIZE USR 63919. El borde de pantalla se pondrá intermitente. Pon tu cinta al principio de un programa y pulsa PLAY. Aparecerá entonces la información que haya en la cabecera de la cinta.

Para hacer la conversión de datos de tipo decimal a hexadecimal, teclea RANDOMIZE USR 64394, mientras que para hacer la conversión de hexadecimal a decimal tienes que teclear RANDOMIZE USR 64453. (En esta última rutina no hay necesidad de pulsar ENTER a continuación del número hexadecimal.)

```
5 CLEAR 63488: BORDER 0:
  PAPER 0: INK 6: CLS
10 PRINT AT 0,9; INVERSE 1;
   "HERRAMIENTAS"
12 PRINT AT 8,2;" PULSA UNA
   TECLA PARA CARGAR  EL
   CODIGO MAQUINA DE
   REFERENCIASCRUZADAS.":
   PAUSE 0
14 LOAD "CREF"CODE
15 CLS : PRINT "POKEANDO EL
   CODIGO MAQUINA DE[3*
   ESPACIO]HERRAMIENTAS.
   PREPARA UN CASSETTEPARA
   SALVAR."
20 LET L=100: RESTORE L: FOR
   N=63489 TO 64560 STEP 16
30 LET T=0: FOR D=0 TO 15
40 READ A: POKE (N+D),A: LET
   T=T+A: NEXT D
50 READ A:IF A<>T THEN PRINT
   "ERROR EN LINEA ";L: STOP
60 LET L=L+10
70 NEXT N
100 DATA 62,12,205,48,252,
    205,60,250,237,67,155,
    248,42,83,92,1,2019
110 DATA 0,0,126,254,128,40,
    9,197,205,184,25,193,3,
    235,24,242,1865
120 DATA 205,43,45,58,155,
    248,205,40,45,239,4,56,
    205,162,45,33,1788
130 DATA 15,39,167,237,66,48
    ,2,207,5,33,145,248,126,
    60,40,32,1470
140 DATA 35,229,237,91,83,92
    ,42,75,92,167,237,82,68,
    77,235,237,2079
150 DATA 177,197,229,245,204
    ,157,248,241,225,193,234
    ,80,248,225,24,220,3147
160 DATA 42,83,92,58,155,248
    ,54,0,35,119,205,40,45,
    239,49,192,1656
170 DATA 56,42,83,92,205,184
    ,25,42,75,92,43,167,237,
    82,216,235,1876
180 DATA 229,239,224,15,49,
    56,205,162,45,225,112,35
    ,113,43,24,228,2004
190 DATA 201,224,228,235,236
    ,239,246,255,0,0,10,0,
    229,6,4,35,2148
200 DATA 126,254,14,40,4,16,
    248,225,201,197,35,35,35
    ,78,35,70,1613
210 DATA 42,83,92,217,1,1,0,
    217,205,149,22,43,235,
    167,237,66,1777
220 DATA 235,48,11,217,3,217
    ,197,205,184,25,193,235,
    24,234,217,197,2442
230 DATA 217,209,42,155,248,
    205,169,48,235,42,104,92
    ,35,35,115,35,1986
240 DATA 114,235,62,0,167,1,
    9,0,237,66,56,17,60,1,90
    ,0,1115
```

```
250 DATA 237,66,56,9,60,1,
    132,3,237,66,56,1,60,209
    ,225,229,1647
260 DATA 245,130,214,4,245,6
    ,0,56,9,79,40,12,205,85
    ,22,35,1387
270 DATA 24,6,237,68,79,205,
    232,25,193,241,197,79,6,
    0,9,65,1666
280 DATA 229,197,35,35,235,
    42,104,92,1,5,0,237,176,
    193,225,229,2035
290 DATA 197,239,224,164,5,
    58,193,164,4,224,1,3,225
    ,192,2,56,1951
300 DATA 205,213,45,193,225,
    198,48,119,43,16,228,229
    ,42,104,92,35,2035
310 DATA 35,126,225,198,48,
    119,241,193,245,42,83,92
    ,167,229,237,66,2346
320 DATA 225,48,8,197,205,
    184,25,193,235,24,242,
    235,35,35,193,126,2210
330 DATA 128,119,201,62,0,
    205,48,252,205,26,31,33,
    0,0,62,0,1372
340 DATA 237,66,229,193,205,
    43,45,62,254,205,1,22,
    205,227,45,201,2240
350 DATA 42,75,92,237,75,83,
    92,62,0,237,66,229,62,1,
    205,48,1606
360 DATA 252,193,205,43,45,
    205,227,45,62,2,205,48,
    252,201,221,33,2239
370 DATA 32,255,17,17,0,175,
    55,205,86,5,62,3,205,48,
    252,221,1638
380 DATA 33,32,255,221,126,0
    ,198,6,221,229,205,48,
    252,62,13,215,2116
390 DATA 62,4,205,48,252,221
    ,225,221,35,221,126,0,
    254,255,40,10,2179
400 DATA 6,10,221,126,0,221,
    35,215,16,248,62,13,215,
    62,5,205,1660
410 DATA 48,252,221,33,32,
    255,221,78,11,221,70,12,
    195,163,249,205,2266
420 DATA 142,250,62,10,205,
    48,252,205,60,250,205,
    110,25,229,62,13,2128
430 DATA 215,62,11,205,48,
    252,205,60,250,205,110,
    25,193,32,16,229,2118
440 DATA 35,35,126,35,95,126
    ,87,225,237,90,17,4,0,
    237,90,229,1668
450 DATA 197,62,0,237,66,218
    ,87,252,195,89,252,6,0,
    197,205,95,2158
460 DATA 252,205,115,252,193
    ,254,13,40,26,254,58,48,
    240,214,48,56,2268
470 DATA 236,245,4,120,254,6
    ,32,4,5,241,24,225,241,
    245,198,48,2128
480 DATA 215,24,218,221,33,
    49,255,120,254,0,40,207,
    33,0,0,221,1890
490 DATA 94,0,221,86,1,241,
    254,0,40,7,237,90,56,13,
    61,32,1433
500 DATA 249,221,35,221,35,5
    ,32,231,229,193,201,207,
    5,42,75,92,2073
510 DATA 237,75,83,92,237,66
    ,192,207,9,62,10,205,48,
    252,205,60,2040
520 DATA 250,34,30,255,62,13
    ,215,62,12,205,48,252,
    205,60,250,34,1987
530 DATA 28,255,33,48,48,34,
    59,255,34,61,255,237,75,
    30,255,205,1912
540 DATA 115,251,62,2,50,107
    ,92,50,107,92,205,149,23
    ,205,176,22,1708
550 DATA 62,0,205,1,22,33,59
    ,255,6,4,126,229,197,205
    ,129,15,1548
560 DATA 193,225,35,16,245,
    205,44,15,205,23,27,221,
    33,58,92,221,1858
570 DATA 203,0,126,32,13,42,
    89,92,205,167,17,62,255,
    50,58,92,1503
580 DATA 24,206,42,89,92,34,
    93,92,205,251,25,120,177
    ,32,10,223,1715
590 DATA 254,13,40,174,205,
    176,22,207,1,237,67,73,
    92,42,93,92,1788
600 DATA 235,33,85,21,229,42
    ,97,92,55,237,82,229,96,
    105,205,110,1953
610 DATA 25,32,6,205,184,25,
    205,232,25,193,121,61,
    176,40,47,197,1774
620 DATA 3,3,3,3,43,237,91,
    83,92,213,205,85,22,225
    ,34,83,1425
630 DATA 92,193,197,19,42,97
    ,92,43,43,237,184,42,73,
    92,235,193,1874
640 DATA 112,43,113,43,115,
    43,114,237,75,28,255,205
    ,115,251,241,195,2185
650 DATA 195,250,33,62,255,
    126,60,254,58,40,8,119,
    11,121,128,176,1896
660 DATA 200,24,239,62,48,
    119,43,24,236,62,14,205,
    48,252,205,60,1841
670 DATA 250,62,13,215,197,
    62,15,205,48,252,193,46,
    2,96,124,203,1983
680 DATA 31,203,31,203,31,
    203,31,230,15,205,189,
    251,215,124,230,15,2207
690 DATA 205,189,251,215,97,
    45,32,230,62,13,215,201,
    198,48,254,58,2313
700 DATA 216,198,7,201,62,16
    ,205,48,252,17,85,255,6,
    4,213,197,1982
710 DATA 205,95,252,205,115,
    252,215,245,241,193,209,
    18,19,16,239,62,2581
720 DATA 13,215,62,17,205,48
    ,252,221,33,85,255,17,0,
    16,33,0,1472
730 DATA 0,14,4,221,126,0,
    221,35,214,48,218,87,252
    ,254,10,56,1760
740 DATA 2,214,7,254,16,210,
    87,252,71,254,0,40,3,25,
    16,253,1704
750 DATA 203,58,203,27,203,
    58,203,27,203,58,203,27,
    203,58,203,27,1964
760 DATA 13,32,208,229,193,
    205,43,45,205,227,45,62,
    13,215,201,203,2139
800 CLS : PRINT AT 5,5;
    " COMPILACION
    COMPLETA."
810 PRINT AT 7,1;"PREPARA UN
    CASSETTE PARA SALVAR"
820 PRINT AT 9,4;"FICHERO"
    "HERRAM."" CODE"
830 SAVE "HERRAM."CODE 63489
    ,2000
```



RUTINAS EN MEMORIAS ROM (I)

Muchos ordenadores domésticos, entre los cuales se encuentra lógicamente el Spectrum, **tienen como microprocesador (C.P.U.) al** Z-80A, **el cual puede direccionar hasta 65535 posiciones de memoria, es decir 64 K (1K = 1024) a través de sus 16 líneas del Bus de Direcciones.**

De los 64 K de memoria del **Spectrum**, 16 K son ocupados por la memoria ROM (*Read Only Memory*) y 48 K por la memoria RAM (*Random Access Memory*).

Nos centraremos solamente en la zona de la ROM, que son físicamente 8 circuitos integrados de 2 Kbytes cada uno y en la organización de la memoria abarcan las posiciones 0000-16383, en decimal, ó $0000-$3FFF en hexadecimal.

Los 16 K de memoria se dividen a su vez en:

EL SISTEMA OPERATIVO, posiciones 0000-7405 ($0000-$1CED)

EL INTERPRETE BASIC, posiciones 7406-15615 ($1CEE-$3CFF)

EL JUEGO DE CARACTERES, posiciones 15616-16383 ($3D00-$3FFF).

En esta primera parte, trataremos solamente el Sistema Operativo (S.O.). Posteriormente incluiremos también el Intérprete BASIC y el juego de caracteres.

Al final de las rutinas y tablas del Sistema Operativo, incluiremos algunos ejemplos de uso de las rutinas de la ROM, tanto desde el BASIC como desde el Código Máquina.

SISTEMA OPERATIVO

Ciento ochenta y dos son las rutinas y tablas del S.O. del **Spectrum**.

RUTINAS DE RESTAURACION:

00-07 ($0000-$0007).- «RST 0000» Rutina de inicialización llama a la rutina de 4555 para inicializar la memoria, pantalla y variables, START.

08-15 ($0008-$000F).- «RST 0010» Rutina de tratamiento de errores, el octeto siguiente da el tipo de error, retorna al BASIC.

16-23 ($0010-$0017).- «RST 0010» Rutina de presentación de un caracter en pantalla, los datos y el caracter en el acumulador.

24-39 ($0018-$0027).- «RST 0018» y «RST 0020» Rutinas que situan en el acumulador un caracter a intrepretar por el BASIC.

40-47 ($0028-$002F).- «RST 0028» Rutina de cálculo en coma flotante, con un salto a la 13147 (el Calculador).

48-55 ($0030-$0037).- «RST 0030» Rutina para reservar espacio, para las variables, el contenido de BC, MAKE BC-SPACE.

56-82 ($0038-$0052).- Rutina de tratamiento de las interrupciones enmascarables.

83-101 ($0053-$0065).- Rutina de tratamiento de error, ERROR-2.

102-115 ($0066-$0073).- Rutina de tratamiento de las interrupciones NO-enmascarables.

116-124 ($0074-$007C).- Rutina para incrementar el puntero CH-EDD.

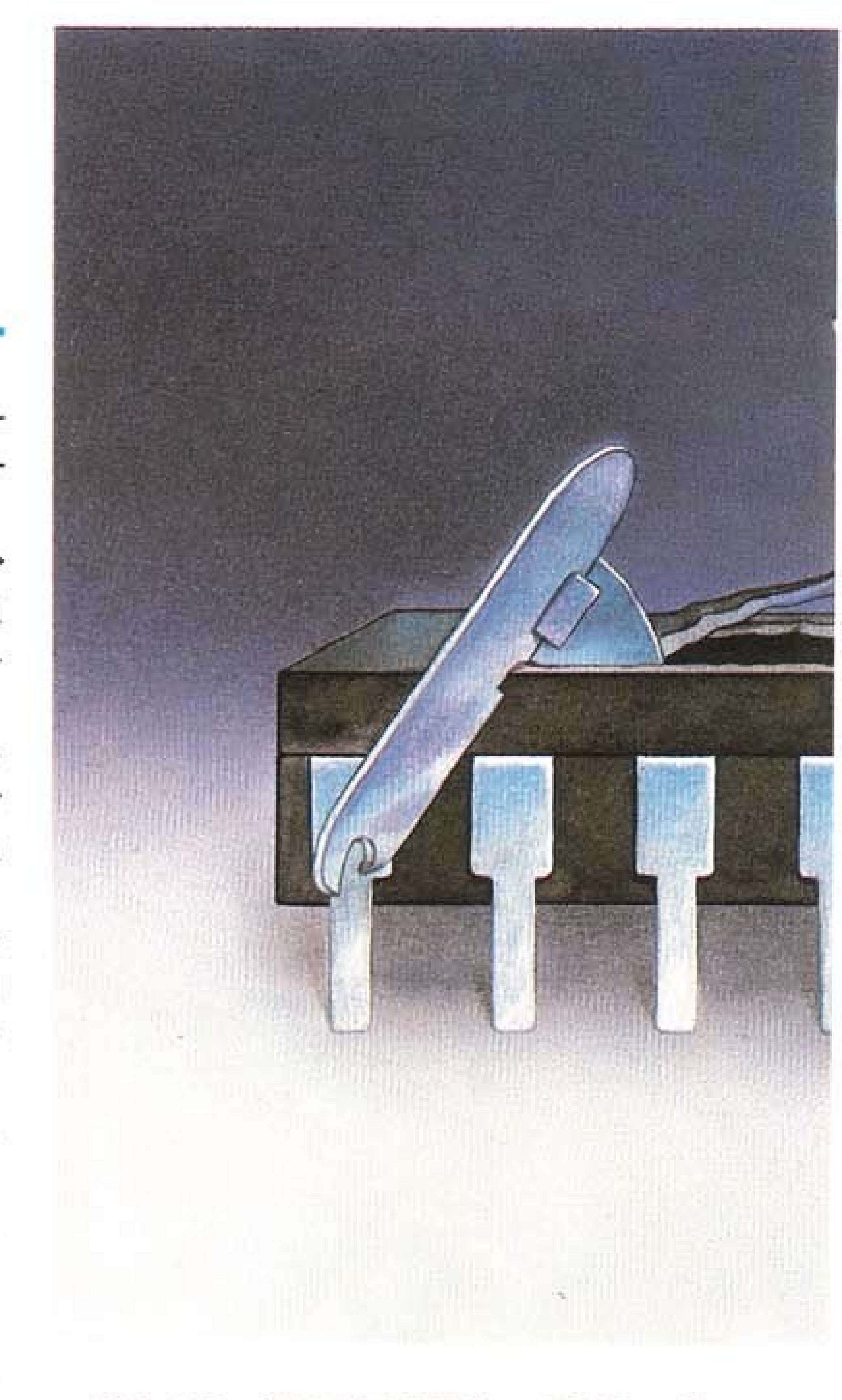
125-148 ($007D-$0094).- Rutina que testea el contenido del acumulador, llamada por RST 0018, SKIP-OVER.

149-516 ($0095-$0204).- Tabla de las instrucciones usadas por el BASIC. (Lista de Códigos).

517-555 ($0205-$022B).- Tabla de las teclas en modo L (LETRAS).

556-581 ($022C-$0245).- Tabla de las funciones en modo K (KEYWORDS).

582-607 ($0246-$025F).- Tabla de las funciones en modo E (EXTENDIDO).



608-617 ($0260-$0269).- Tabla de los códigos de CONTROL.

618-643 ($026A-$0283).- Tabla de las funciones en modo K con SYMBOL SHIFT .

644-653 ($0284-$028D).- Tabla de las funciones en modo E con SYMBOL SHIFT

RUTINAS DE CONTROL DEL TECLADO

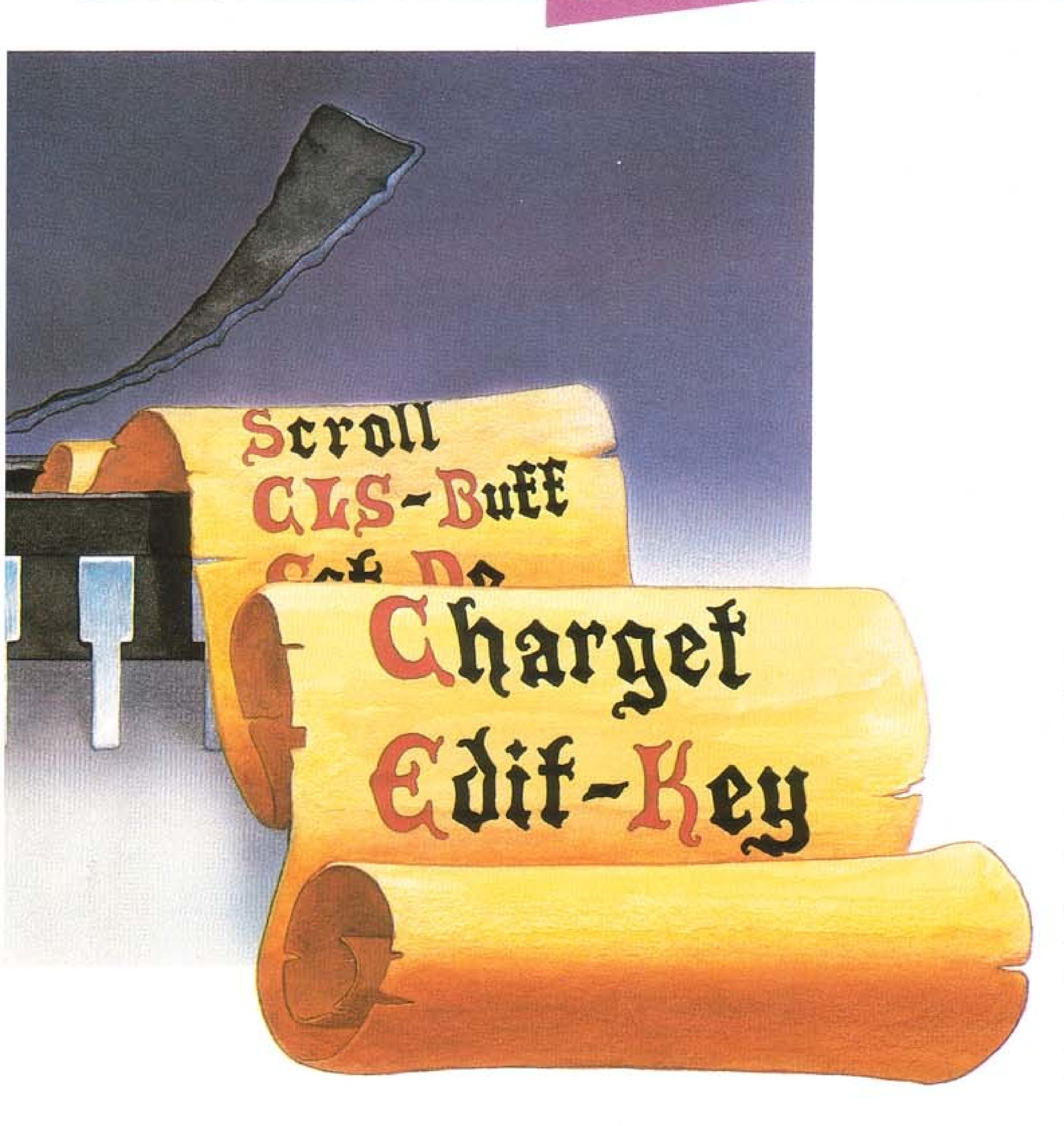
654-702 ($028E-$02BE).- Rutina de exploración del TECLADO. KEY-SCAN.

703-783 ($02BF-$030F).- Rutina de tratamiento del teclado llamada cada 20 ms. por la interrupción enmascarable, KEYBOARD.

784-797 ($0310-$031D).- Rutina de repetición de una tecla. K-REPEAT.

798-818 ($031E-$0332).- Rutina de tratamiento de comprobación de tecla, REPETICION, SONIDO, VELOCIDAD, etc..., K-TEST.

819-948 ($0333-$03B4).- Rutina de decodificación del código de la tecla



pulsada, almacena el código en el acumulador. K-DECODE.

RUTINAS DE CONTROL DEL SONIDO

949-1015 ($03B5-$03F7).- Rutina de ejecución de un sonido simple, el registro DE con los valores de la frecuencia y tiempo y el registro HL el tiempo de ejecución, BEEPER.

1016-1133 ($03F8-$046D).- Rutina del comando BEEP.

1134-1193 ($046E-$04A9).- Tabla de las frecuencias de los doce semitonos de una octava.

VARIOS ZX81

1194-1217 ($04AA-$04C1).- Rutinas de error aplicables al ordenador **ZX81**. (Tener en cuenta que el sistema operativo del **Spectrum**, es una adaptación del **ZX81**).

RUTINAS DE CONTROL DEL CASSETTE LOAD / SAVE / VERIFY / MERGE

1218-1342 ($04C2-$053E).- Rutina para almacenar las informaciones de ficheros, SA-BYTES.

1343-1365 ($053F-$0555).- Rutina de cambio del BORDE, para tratamiento en instrucciones de LOAD y SAVE, SAL/LD-RET.

1366-1506 ($0556-$05E2).- Rutina de carga para la instrucción LOAD o VERIFY, LD-BYTES.

1507-1540 ($05E3-$0604).- Rutina de comprobación de la cabecera en la carga, LD-EDGE-2.

1541-1887 ($0605-$075F).- Rutina del tratamiento de las instrucciones SAVE, LOAD, VERIFY, MERGE.

1888-1994 ($0760-$07CA).- Rutina de tratamiento de carga para comprobar la cabecera y el tiempo de separación, con el bloque de datos.

1995-2049 ($07CB-$0801).- Rutina de control de la instrucción VERIFY.

2050-2055 ($0802-$0807).- Rutina de carga de un bloque de bytes, en las instrucciones LOAD, VERIFY, MERGE.

2056-2229 ($0808,$08B5).- Rutina de control de la instrucción LOAD.

2230-2347 ($08B6-$092B).- Rutina de control de la instrucción MERGE.

2348-2415 ($092C-$096F).- Rutina de carga de una línea BASIC o de una variable.

2416-2464 ($0970-$09A0).- Rutina de control de la instrucción SAVE.

2465-2547 ($09A1-$09F3).- Relación de códigos de los mensajes del *cassette*.

RUTINAS DE CONTROL DE IMPRESION PANTALLA / IMPRESORA

2548-2576 ($09F4-$0A10).- Rutina de inicialización, el registro A contiene el caracter a representar, PRINT-OUT.

2577-2594 ($0A11-$0A22).- Tabla de los códigos de los caracteres de CONTROL.

2595-2620 ($0A23-$0A3C).- Rutina del tratamiento del CURSOR en desplazamiento a la izquierda.

2621-2638 ($0A3D-$0A4E).- Rutina del tratamiento del CURSOR en desplazamiento a la derecha.

2639-2654 ($0A4F-$0A5E).- Rutina de retorno del CARRO, en fase de tratamiento con la impresora.

2655-2664 ($0A5F-$0A68).- Rutina de tratamiento de presentación de la COMA.

2665-2668 ($0A69-$0A6C).- Rutina de tratamiento de presentación del signo de INTERROGACION.

2669-2776 ($0A6D-$0AD8).- Rutina del tratamiento de los caracteres de CONTROL, con operandos.

2777-2779 ($0AD9-$0ADB).- Rutinas del tratamiento de los caracteres «presentables», PO-ABLE.

2780-2818 ($0ADC-$0B02).- Rutina de modificación del contenido de las variables, con el nuevo valor del cursor. (línea/columna).

2819-2851 ($0B03-$0B23).- Rutina de búsqueda del valor del cursor, en las variables del sistema.

2852-2942 ($0B24-$0B7E).- Rutina de presentación de un caracter.

2943-3034 ($0B7F-$0BDA).- Presenta todos los caracteres en una matriz de 8 × 8.

3035-3081 ($0BDB-$0C09).- Rutina de búsqueda del atributo, en las variables del sistema. (ATTR-T, MASK-T, P-FLAG).

3082-3130 ($0C0A-$0C40).- Rutina de presentación de un mensaje, el registro A contiene la longitud, el DE la dirección.

3131-3136 ($0C3B-$0C40).- Rutina de presentación de un caracter, salvando los registros, PO-SAVE.

3137-3156 ($0C41-$0C54).- Situa en el registro DE la dirección de la tabla a presentar.

3157-3319 ($0C55-$0CF7).- Rutina de tratamiento de comprobación de la instrucción SCROLL.

3320-3404 ($0CF8-$0D4C).- Tabla conteniendo los códigos de los caracteres del mensaje «*scroll*».

3405-3434 ($0D4D-$0D6A).- Rutina del almacenamiento temporal del atributo, el color, en la variable ATTR-T, MASK-T.

3435-3502 ($0D6B-$0DAE).- Rutina de la instrucción CLS, tratando los atributos de la variable ATTR-P.

3503-3544 ($0D4F-$0DD8).- Rutina de borrado de la zona de presentación, es llamada por la rutina CLS.

3545-3581 ($0DD9-$0DFD).- Indica la dirección del caracter cuyo número de línea y columna está en el registro BC, para tratar con la impresora, cargándolo en el *buffer*, CL-SET.

3582-3651 ($ODFE-$0E43).- Rutina de SCROLL, el número de líneas en el registro B ($17).

3652-3719 ($0E44-$0E87).- Rutina de borrado de líneas, borra el número de líneas contenido en el registro B, comenzando por la parte inferior.

3720-3788 ($0E88-$0E9A).- Almacena en el registro DE, la dirección del atributo apuntado por el registro HL, CL-ATTR.

3739-3755 ($0E9B-$0EAB).- Almacena en el registro HL, la dirección del fichero de presentación del número de líneas contenidas en el registro B, CL-ADDR.

3756-3788 ($0EAC-$0ECC).- Rutina de la instrucción COPY, copia de 176 líneas.

3789-3806 ($0ECD-$0EDE).- Almacena la línea a imprimir en el *buffer* de impresora, primero los 256 bytes a cero, 23296-23551 ($5BOO-$5BFF), más tarde los almacena, COPY-BUFF.

3807-3827 ($0EDF-$0EF3).- Borra el contenido del *buffer* de la impresora, CLS-BUFF, (256 bytes).

3828-3883 ($0EF4-$0F2B).- Rutina de copia de una línea a imprimir, COPY-LINE.

RUTINAS DE TRATAMIENTO DEL PROGRAMA EN EDICION

3884-3968 ($0F2C-$0F80).- Rutina del EDITOR, se utiliza para la entrada de una línea BASIC y para INPUT, en modo EDICION.

3969-3999 ($0F81-$0F9F).- Rutina de inclusión de un caracter en línea BASIC o INPUT, ADD-CHAR.

4000-4008 ($0FA0-$0FA8).- Tabla de caracteres de las teclas utilizadas en modo EDICION.

4009-4082 ($0FA9-$0FF2).- Rutina de tratamiento de la tecla EDIT, EDIT-KEY.

4083-4102 ($0FF3-$1006).- Rutina de tratamiento de la tecla del CURSOR hacia abajo, en modo EDICION, ED-DOWN.

4103-4107 ($1007-$100B).- Rutina de tratamiento de la tecla del CURSOR hacia la izquierda, en modo EDICION, ED-LEFT.

4108-4116 ($100C-$1014).- Rutina de tratamiento de la tecla del CURSOR hacia la derecha, en modo EDICION, ED-RIGHT.

4117-4125 ($1015-$101D).- Rutina de tratamiento de la tecla DELETE, en modo EDICION, ED-DELETE.

4126-4131 ($101E-$1023).- Control de la pulsación del teclado, comprobando dos veces la pulsación de una tecla, antes de la edición, ED-IGNORE.

4132-4144 ($1024-$1030).- Rutina de tratamiento de la tecla ENTER, en modo EDICION, ED-ENTER.

4145-4184 ($1031-$1058).- Rutina para decrementar el registro HL que contiene la dirección del cursor, hasta el principio de la línea, ED-EDGE.

4185-4213 ($1059-$1075).- Rutina de tratamiento de la tecla del CURSOR hacia arriba, en modo EDICION, ED-AP.

4214-4222 ($1076-$107E).- Rutina de tratamiento de la tecla SYMBOL o GRAPHICS, en modo EDICION, ED-SYMBOL.

4223-4246 ($107F-$1096).- Rutina del tratamiento de error en modo EDICION, ED-ERROR.

4247-4263 ($1097-$10A7).- Rutina de borrado, en modo EDICION, CLEAR-SP.

4264-4380 ($10A8-$111C).- Almacena y comunica el valor de la última tecla pulsada, KEY-INPUT.

4381-4495 ($111D-$118F).- Transfiere la línea para editar en la parte inferior de la pantalla, ED-COPY.

4496-4518 ($1190-$11A6).- Almacena en el registro HL la dirección de la memoria de editor, y en el registro DE la dirección final, SET-HL / SET-DE.

4519-4534 ($11A7-$11B6).- Almacena en la pila todos los números en coma flotante de una línea BASIC, REMOVE-FP.

RUTINAS GENERALES DE EJECUCION

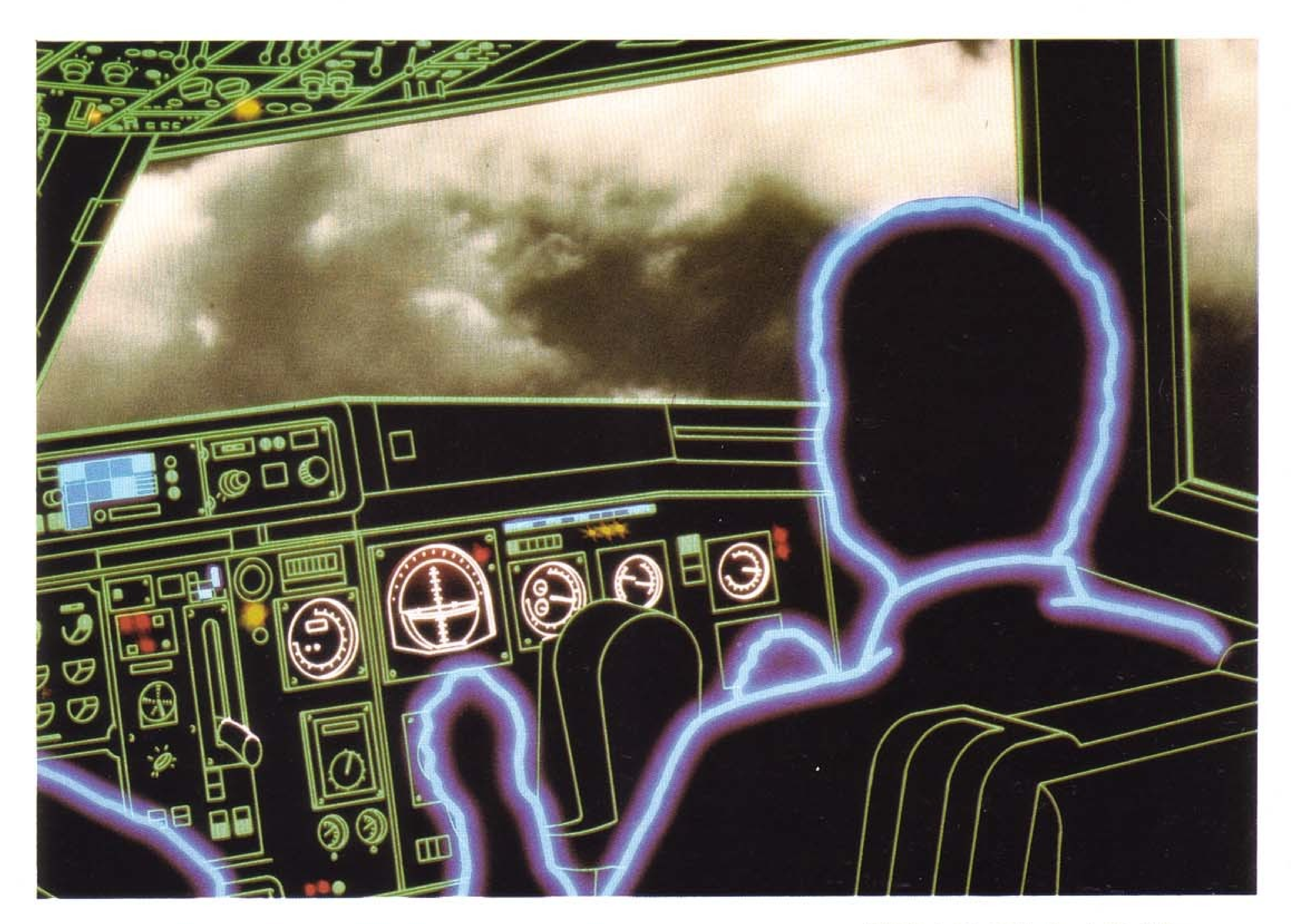
4535-4554 ($11B7-$11CA).- Rutina de tratamiento de la instrucción NEW.

4555-4559 ($11CB-$11CF).- Rutina que coloca el BORDE de color blanco, START/NEW.

4560-4569 ($11D0-$11D9).- Rutina de inicialización del registro indexado I, se inicializa con ($3F), se usa en la generación de (INCOMPLETO).......

LA APROXIMACION FINAL

- RUTINA DE CONTROL DEL TECLADO
- COMPROBACION DE ATERRIZAJE SOBRE LA PISTA
- CHOQUES
- ESTIMACION DE DAÑOS



Quita el piloto automático, pero no exhales todavía el suspiro de alivio: aún tienes los mandos. Utilizando únicamente seis teclas debes conseguir que el avión se pose suavemente sobre la tierra

En esta tercera y última parte del programa simulador de vuelo tendrás por fin el control del aparato.

Hay tres pares de teclas que te permiten aumentar y disminuir la velocidad del motor, hacer que el avión vire o que suba y baje.

Te encontrarás situado a 20000 metros de la pista de aterrizaje y a 2000 metros de altura. No es fácil conducir un avión, en especial si eres un piloto inexperto. No esperes convertirte en experto con unos cuantos vuelos, la capacidad de pilotar un avión es una habilidad que muy pocos adquieren en poco tiempo y con facilidad.

Cuando vayas perseverando y adquiriendo más experiencia en el control de la aeronave, tus maniobras de aterrizaje irán mejorando. Después de cada aterrizaje (o cada accidente) recibirás un informe con los detalles finales del vuelo. Estúdialo para ver qué es lo que has hecho mal, e intenta corregir las deficiencias.

```
3000 LET POW=0: LET K$=
     INKEY$: IF K$="" THEN
     RETURN
3010 IF K$="S" THEN  LET
     POW=-1
3020 IF K$="F" THEN  LET
     POW=1
3030 IF K$="Q" THEN  LET PT=
     PT+1
3040 IF K$="A" THEN  LET PT=
     PT-1
3050 IF K$="O" AND RL>-30
     THEN  LET RL=RL-1
3060 IF K$="P" AND RL<30
     THEN  LET RL=RL+1
3070 RETURN
```

```
5020 CLS : INPUT "VELOCIDAD
     DEL VIENTO (1-50) MPH",
     X0
5025 IF X0>50 OR X0<1 THEN
     GO TO 5020
5030 INPUT "DIRECCION DEL
     VIENTO (0-359) GRADOS",
     X1
5035 IF X1>359 OR X1<0 THEN
     GO TO 5030
5040 LET X0=X0/3
5050 PRINT "VELOC VIENTO= ";
     3*X0;" M/S": PRINT
     "DIRECCION= ";X1;
     " GRADOS"
5055 PAUSE 100: CLS
5060 LET WY=-X0*COS (X1*C)
5070 LET WX=-X0*SIN (X1*C)
5500 GO SUB 3000: GO SUB 1000
5510 IF PZ<=0 THEN GO TO 6000
5520 GO SUB 2000
5530 GO TO 5500
6000 IF ABS RL>RT OR PT>TP
     OR PT<0 OR AS>80 THEN
     GO TO 6030
6010 IF ABS PX>WR OR ABS PY>
     1000 THEN  GO TO 6060
6020 CLS : PRINT
     " FELICIDADES, LOGRO
     ATERRIZAR": GO TO 6100
6030 FOR N=0 TO 20 STEP .5:
     PLOT 127,130: DRAW 120-
     INT (RND*240),45-INT
     (RND*90): BEEP .005,20-
     N: NEXT N
6040 PAUSE 50
6050 CLS : PRINT "CON ESTE
     ACCIDENTE HA DESTRUIDOLA
     NAVE Y MATADO A SUS
     PASAJEROS": GO TO 6100
6060 CLS : PRINT "ATERRIZO
     FUERA DE LA PISTA"
6070 IF AS<40 THEN  PRINT
     "AFORTUNADAMENTE NO IBA
     TAN[6*ESPACIO]RAPIDO
     COMO PARA SUFRIR[9*
     ESPACIO]DESPERFECTOS
     GRAVES": GO TO 6100
6080 IF AS<80 THEN  PRINT "A
     ESTA VELOCIDAD HA
     RESULTADO[3*ESPACIO]CON
     ALGUNOS DESPERFECTOS Y
     [6*ESPACIO]LESIONES
     LEVES": GO TO 6100
6090 PRINT "PERDIDA LA PISTA
     CON ESTA VELOCIDAD A LA
     IZQUIERDA  NO HAY
     SUPERVIVIENTES!"
```



```
6100 PRINT "[7*ESPACIO]
     DETALLES DEL VUELO"
6110 PRINT "VEL.AIR= ";INT
     AS;" M/S": PRINT
     "DISTANCIA= ";INT (SQR
     (PY*PY+PX*PX)): PRINT
     "CAIDA[3*ESPACIO]= ";PT
6120 PRINT "BALANCEO = ";RL:
     PRINT "RPM[5*ESPACIO]=
     ";INT (10*TC)/10;" X
     1000"
6130 PRINT "DERIV[3*ESPACIO]
     = ";INT ABS PX;" MTRS":
     PRINT "RUMBO = ";AD;
     " GRADOS"
6140 PRINT '"QUIERE JUGAR
     OTRA VEZ (S/N)?"
6150 LET A$=INKEY$: IF A$<>
     "S" AND A$<>"N" THEN
     GO TO 6150
6160 IF A$="N" THEN  CLS :
     STOP
6170 RUN
```

La primera sección del programa, que es la subrutina que se extiende entre las líneas 3000 y 3070, es la que te da el control sobre el aeroplano.

Utilizando la tecla INKEY$ para detectar las pulsaciones de tecla, puedes cambiar las revoluciones del motor, la elevación y la posición del avión. Se utiliza una nueva variable, POW, en las líneas 3010 y 3020, para cambiar las revoluciones del motor; consulta las líneas 2225 y 2226 de la página 102 (INPUT n.° 13). Cada vez que se llama a la subrutina, se pone POW a cero en la línea 3000. La tecla S se utiliza para disminuir la velocidad del motor, mientras que para aumentarla se utiliza F.

Para cambiar el ángulo de inclinación se utilizan Q y A, que incrementan o decrementan la variable PT de las líneas 3030 y 3040. Finalmente, puedes utilizar O y P para ladear el avión. De esto se ocupan las líneas 3050 y 3060, al modificar la variable RL.

Las líneas 5020 a 5070 te permiten modificar el nivel de dificultad del aterrizaje simulado al darte la opción de enfrentarte con vientos de diferentes velocidades y de poder elegir su dirección. La opción más sencilla es un viento con una velocidad de 1 m/s y con una dirección de 0 grados. Partiendo de la elección que hayas hecho, la línea 5060 calcula la velocidad del viento en la dirección de la marcha, mientras que la línea 5070 calcula la velocidad del viento en la dirección perpendicular a la marcha. Para calcular los componentes del vector velocidad en cada dirección, se utilizan las funciones SIN y COS. El programa utiliza WX y WY para modificar la posición del avión: GX y GY en la línea 5080.

El núcleo del programa son las cuatro líneas que van desde la 5500 a la 5530, que son las que van llamando por turno a todas las subrutinas importantes al objeto de ir actualizando la posición del avión; ya tienes introducidas todas estas subrutinas.

La línea 5510 examina si el avión ha tocado el suelo, comprobando el parámetro de altura PZ (¿alcanzó el va-



lor cero?). El programa salta entonces al bucle «aéreo» y entra en la rutina que comprueba si el avión ha tomado tierra con éxito.

La línea 6000 de esta rutina comprueba si el valor del balanceo se encuentra dentro de la tolerancia permitida para el balanceo, si la inclinación está dentro de los límites de tolerancia permitidos, si los cambios de demora no superan los límites de guiñada permitidos y si la velocidad del aire es mayor que 80 m/s. Cuando falla cualquiera de estas condiciones, se produce un accidente. Puede deberse a que ha aterrizado con un ala demasiado baja, con la nariz hacia abajo, en una dirección oblicua con respecto al eje de la pista o, simplemente, con demasiada velocidad. Caso de ocurrir alguna de estas cosas, el programa salta a la rutina de accidente. La línea 6030 dibuja unas grietas que se extienden radialmente desde el centro de la ventana de la cabina. Al dibujarse las grietas, se produce también un BEEP. Después del accidente se produce una pausa por acción de la línea 6040, antes de que la línea 6050 comunique al piloto las malas noticias y le presente todos los detalles del vuelo.

Si el avión no sufre accidente alguno, la línea 6010 comprueba la posición en que ha quedado al aterrizar. Si el aterrizaje se ha producido fuera de la pista, el programa salta a la línea 6060 que se encarga de decirle al piloto lo ocurrido. Si la velocidad del aire está por debajo de los 50 m/s, la línea 6070 informa al piloto de que ha tomado tierra sin percances. Análogamente, las líneas 6080 y 6090 informan si el accidente del avión ha originado únicamente daños ligeros y unas cuantas contusiones o por el contrario ha sido un desastre total sin supervivientes, en cuyo caso puedes empezar a buscar la caja negra.

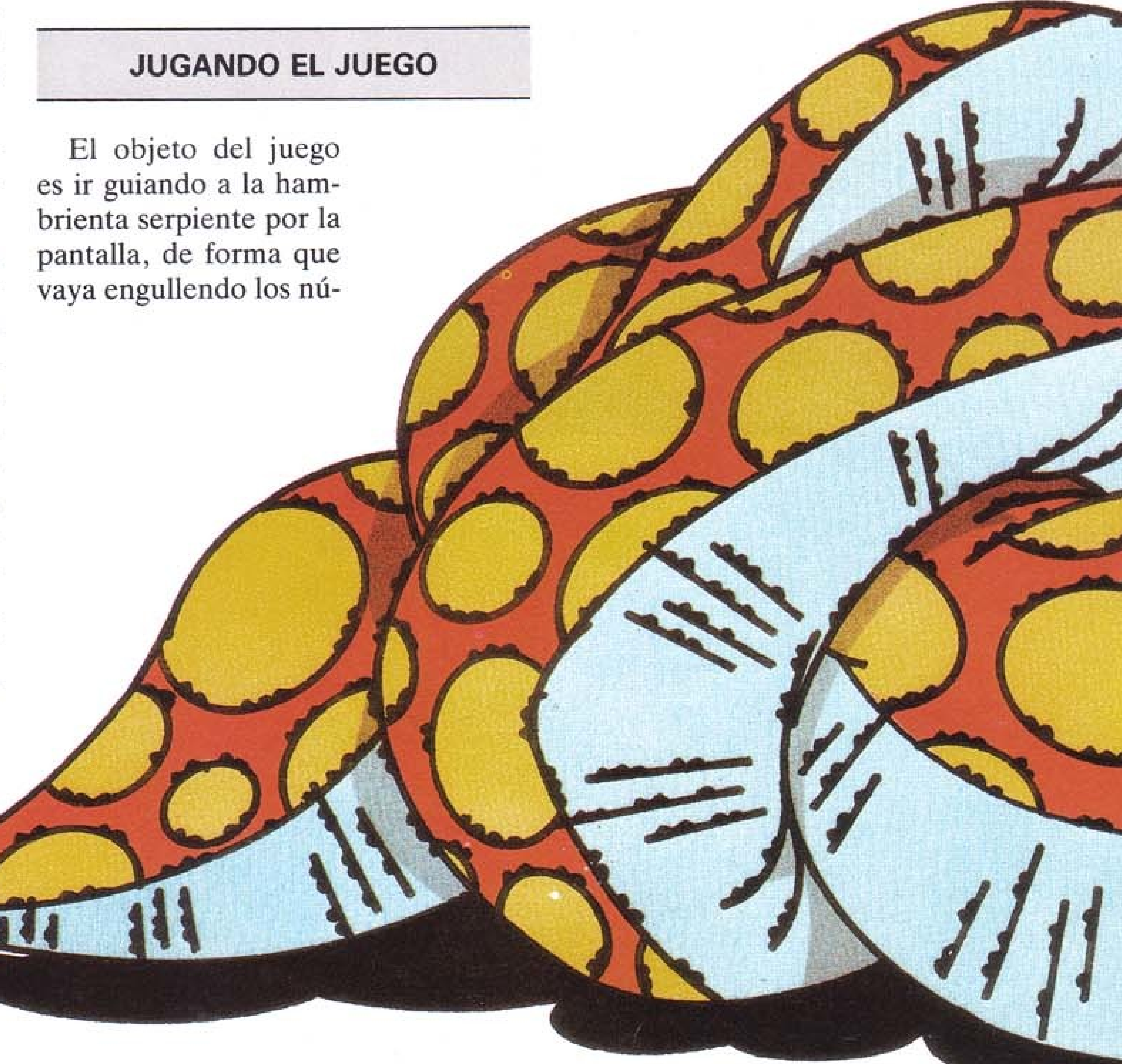
LAS SERPIENTES SUMADORAS

Guía a la serpiente hambrienta para que coma en el «Juego de la serpiente de INPUT». Al engullir ávidamente los suculentos números, la pequeña serpiente irá creciendo hasta hacerse enorme. Naturalmente, siempre que tengas la necesaria destreza.

El juego de la serpiente es uno de los más conocidos y más fáciles de jugar, pese a lo cual continúa siendo enormemente enviciante. Por suerte, no hace falta recurrir al código máquina para programar un juego de este tipo; precisamente este es un juego que ha marcado época en la historia de los ordenadores domésticos, figurando como uno de los más satisfactorios que pueden escribirse en BASIC.

JUGANDO EL JUEGO

El objeto del juego es ir guiando a la hambrienta serpiente por la pantalla, de forma que vaya engullendo los nú-



- EL CLASICO JUEGO EN UNA VERSION PARA BASIC
- DIBUJO DE LA COMIDA
- COMIENDO NUMEROS
- LA CULEBRA SE EXTIENDE

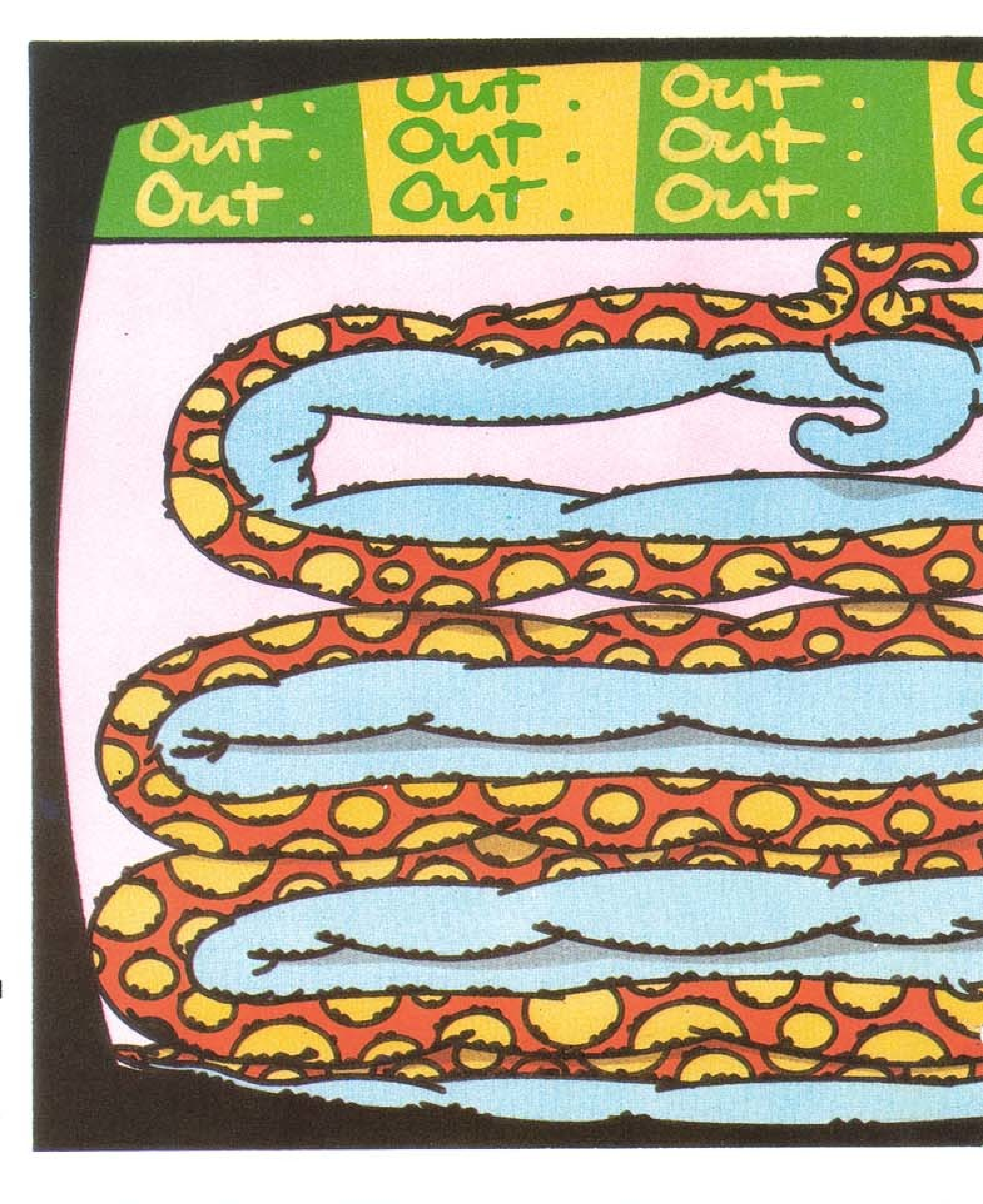


meros que van apareciendo de forma aleatoria. Los números van disminuyendo, de forma que cuanto más tarde la serpiente, más baja será la puntuación. Si tardas demasiado y el número se decrementa hasta cero, desaparecerá y en su lugar aparecerá otro número. Cada vez que la serpiente se traga un número, su longitud aumenta en el correspondiente número de segmentos.

Ten cuidado de no sobrepasar los bordes y de no permitir que la serpiente se entrecruce consigo misma, cosa que te irá resultando más difícil a medida que se va haciendo más larga. Si atraviesas uno de los bordes o el cuerpo de la serpiente, el juego terminará y aparecerá un mensaje de FIN! que cubre toda la pantalla.

```
10 BORDER 1: PAPER 7: INK 9:
   CLS
20 LET HS=0
30 DIM S(570,2)
100 LET S=0
110 LET S(1,1)=10: LET S(1,
    2)=14
120 LET S(2,1)=11: LET S(2,
    2)=14
130 LET S(3,1)=12: LET S(3,
    2)=14
135 GO SUB 1500
140 LET T=1: LET H=3
```

```
145 LET YV=1: LET XV=0
150 FOR N=1 TO 3: PRINT PAPER
    4;AT S(N,1),S(N,2);"#":
    NEXT N
160 LET Y=12: LET X=14
165 LET P=0
170 GO SUB 1000
190 IF ATTR (Y,X)<>56 AND
    ATTR (Y,X)<128 THEN  GO
    TO 2000
200 LET H=H+1: IF H=501 THEN
    LET H=1
210 PRINT PAPER 4;AT Y,X;"#"
220 LET S(H,1)=Y: LET S(H,2)
    =X
230 PRINT AT S(T,1),S(T,2);
    CHR$ 32
240 LET T=T+1: IF T=501 THEN
    LET T=1
250 IF P=0 THEN  LET P=INT
    (RND*9)+1: LET FY=INT
    (RND*19)+2: LET FX=INT
    (RND*30)+1: IF ATTR (FY,
    FX)<>56 THEN  LET P=0:
    GO TO 250
260 PRINT  PAPER INT (P/2);
    FLASH 1;AT FY,FX;P
270 IF RND<.98 THEN  GO TO
    290
280 LET P=P-1: IF P=0 THEN
    PRINT AT FY,FX;CHR$ 32
290 IF Y<>FY OR X<>FX THEN
    GO TO 170
300 LET S=S+P: PRINT  PAPER
    4;AT Y,X;"#": PRINT
    PAPER 6;AT 0,6;S
310 FOR N=1 TO P
320 GO SUB 1000
325 IF ATTR (Y,X)<>56 THEN
    GO TO 2000
330 LET H=H+1: IF H=501 THEN
    LET H=1
340 LET S(H,1)=Y: LET S(H,2)
    =X
350 PRINT  PAPER 4;AT S(H,1)
    ,S(H,2);"#"
355 FOR M=1 TO 10: NEXT M
360 NEXT N
500 GO TO 165
1000 LET A$=INKEY$
1010 IF A$="Q" THEN  LET YV=
     -1: LET XV=0
1020 IF A$="A" THEN  LET YV=
     1: LET XV=0
1030 IF A$="O" THEN  LET XV=
     -1: LET YV=0
1040 IF A$="P" THEN  LET XV=
     1: LET YV=0
1050 LET Y=Y+YV: LET X=X+XV:
     RETURN
1500 FOR N=22560 TO 22591:
     POKE N,16: POKE N+640,
     16: NEXT N
1510 FOR N=22592 TO 22592+
     32*19 STEP 32: POKE N,
     16: POKE N+31,16:NEXT N
1520 PRINT  PAPER 1;AT 0,0;
     "PTOS.  ; PAPER 6;S;
     PAPER 1;TAB 14;"MAX.
     PUNTUACION "; PAPER 6;
     HS; PAPER 1;TAB 31;" "
1590 RETURN
2000 PRINT AT 0,0;: FOR N=1
     TO 88: PRINT  FLASH 1;
     PAPER 2; INK 6;"FIN!:
     BEEP .005,60-N: NEXT N
2010 IF S>HS THEN  LET HS=S
2020 CLS : PRINT AT 8,10;"
     PUNTOS: ";S;AT 11,8;"
     MAX. PUNTUACION: ";HS
2030 PRINT  INVERSE 1;AT 16,
     6;"DESEAS JUGAR OTRA
     VEZ?"
2040 PAUSE 0: CLS : GO TO
     100
```



En la línea 10 se ajustan los bordes, los colores de la tinta y el papel, y se borra toda la pantalla. La puntuación

máxima y la puntuación actual se ponen a cero en las líneas 20 y 100.

La línea 30 dimensiona la matriz s, que se utilizará para almacenar las coordenadas de pantalla correspondientes a la serpiente. Inicialmente la serpiente ocupa las coordenadas (10,14), (11,14) y (12,14). Con las líneas 110 a 130, las coordenadas se sitúan en los primeros tres elementos de la matriz.



En la línea 135 se produce la llamada a la subrutina de «definición de pantalla», la cual comienza en la línea 1500. El borde se dibuja POKEando en el fichero de atributos. La línea 1500 es la encargada de dibujar las líneas superior e inferior de la pantalla, mientras que la línea 1510 dibuja los bordes laterales. En la línea 1520 se asignan los valores de la puntuación máxima y la puntuación actual.

La línea 140 define los punteros de la cabeza y de la cola de la matriz s. Estos punteros son los que le informan a la máquina que las coordenadas de la cola están almacenadas en el primer par de elementos de la matriz y las coordenadas de la cabeza están almacenadas en el primer par de elementos. Cuando la serpiente se mueve, los dos punteros se ajustan de manera que la nueva posición de la cabeza queda cargada en la matriz.

Los vectores xv, yv van siguiendo la dirección en que se mueve la serpiente. Ambos vectores pueden tomar tres valores: el valor 0 significa que la serpiente no está apuntando en esa dirección; el 1 significa que está apuntando hacia abajo o sigue recto, mientras que el valor −1 en cambio significa que la serpiente está apuntando hacia arriba o que sigue recta. En consecuencia la línea 145 hace que al principio del juego la serpiente se mueva por la pantalla hacia arriba.

La serpiente está formada por tres signos # y se construye en la línea 150. La posición de la cabeza en cada momento viene dada por los valores de x e y, utilizándose para detectar las colisiones. Con la línea 160, la cabeza queda posicionada en 14, 12. La línea 160 pone a cero el valor del número presentado p.

En la línea 170 se llama a la subrutina INKEY$, que empieza en la línea 1000. Las letras [Q] y [A] desplazan la serpiente hacia arriba y hacia abajo, mientras que [O] y [P] producen el desplazamiento a derecha e izquierda. Al pulsar las teclas, van cambiando los valores de xv e yv y la línea 1050 modifica la posición de la cabeza, sumando xv e yv a x y v.

Al final de la subrutina, el programa regresa a la línea 190. La línea 190 utiliza la función ATTR para comprobar si la cabeza de la serpiente va a colocarse sobre algún cuadrado que no sea blanco (el color de fondo), y de que no está parpadeando (un número). Si ocurren ambas cosas, la serpiente se ve obligada a superar el borde o entrecruzarse consigo misma, con lo cual el programa salta a la línea 2000, que es la dirección de comienzo de la subrutina de «Final de Juego».

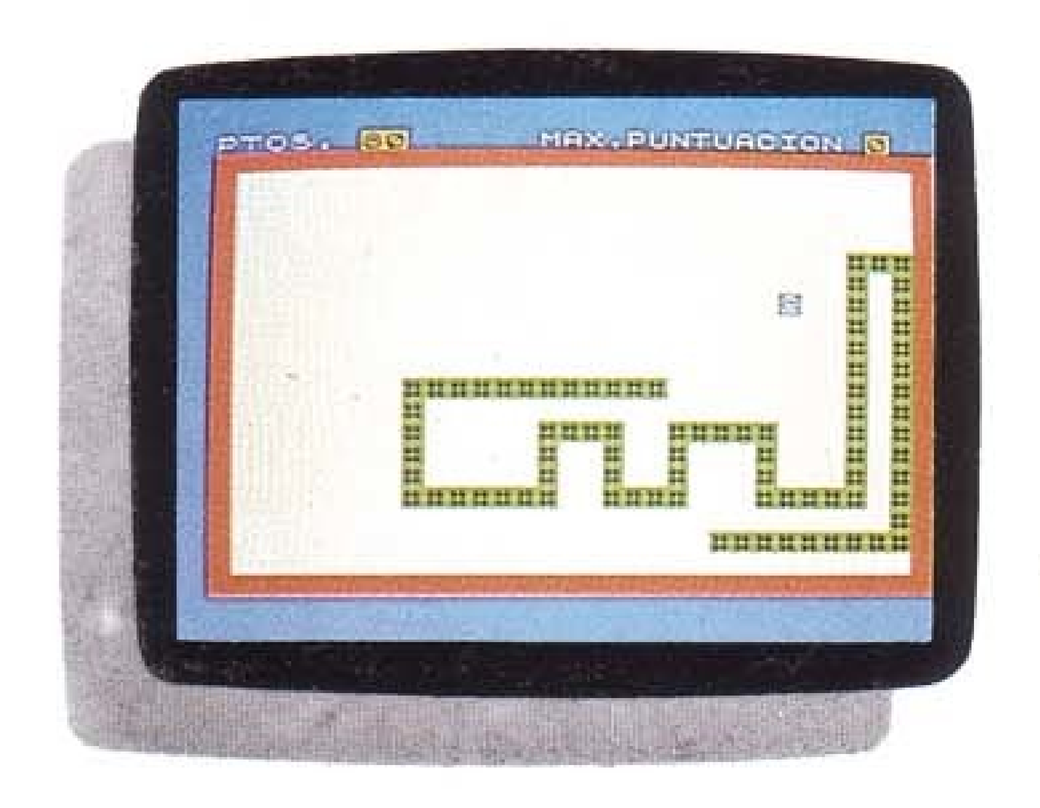
En el supuesto de que la siguiente posición de la cabeza sea legal, la línea 200 incrementa el puntero de la cabeza. Si el puntero resulta tener ahora un valor que supera las dimensiones de la matriz, se resetea a uno. La cabeza se presenta entonces en su nueva posición por medio de la línea 210, mientras que la línea 220 carga la nueva posición en la matriz s, en los elementos indicados por el puntero de la cabeza. Naturalmente, hay que recortar la cola, ya que en caso contrario la serpiente se iría haciendo cada vez más larga; esto se hace en la línea 230, con referencia a la matriz para la posición de la cola. La línea 240 incrementa el puntero de la cola y comprueba de nuevo si dicho puntero está apuntando fuera de las dimensiones de la matriz.

Si no hay número en la pantalla, la línea 250 selecciona un número p y una posición aleatoria para él. Si dicha posición no está situada en una zona blanca de la pantalla, es decir, si ATTR(fy,fx) < > 56, el valor de p se pone a cero y se elige un nuevo número y una nueva posición. Cuando la posición elegida es correcta, la línea 260 hace que el número aparezca intermitente sobre la pantalla.

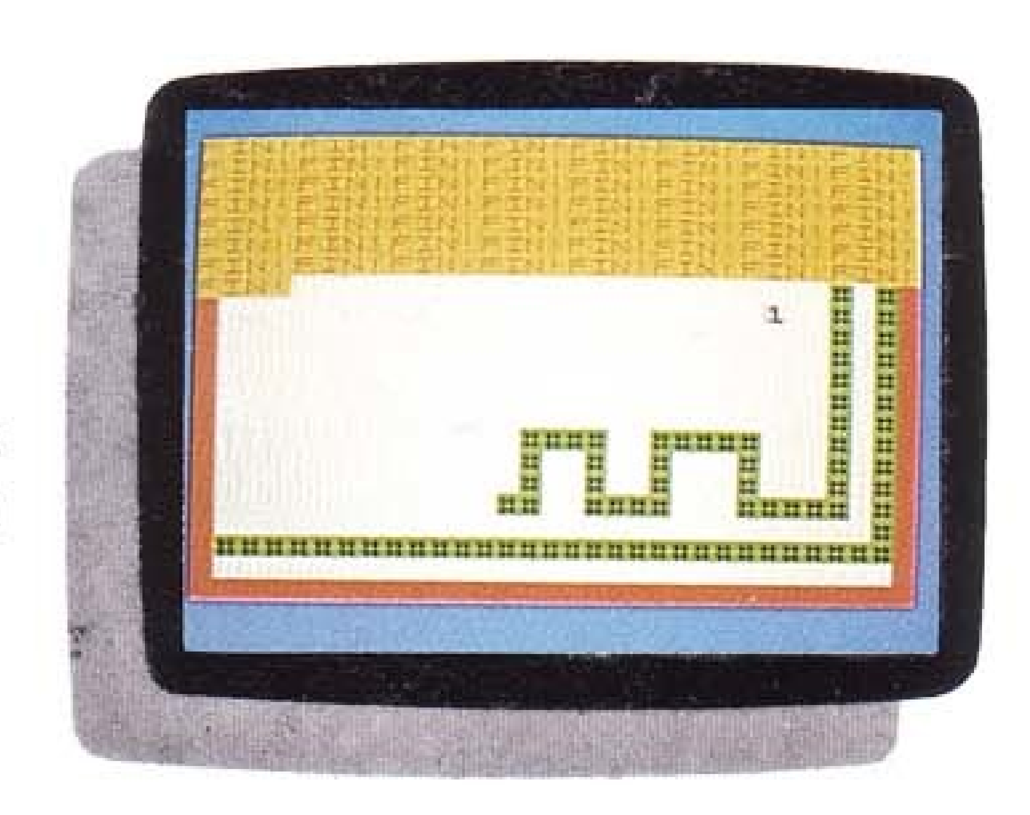
Pasa un cierto tiempo y el número que hay en la pantalla se decrementa. En la línea 270 se introduce un ligero elemento de aleatoriedad, comparando el número aleatorio con el valor 0.98. En la mayoría de los casos, la línea 280, que es la que se encarga de decrementar la puntuación, comprueba que el número continúa siendo mayor que cero después de ser decrementado; en caso contrario, el número es borrado. Si la cabeza no ocupa la misma posición de pantalla que el número, el bucle se completa al ser enviado el programa nuevamente a la línea 170.

Cuando la serpiente se come el número, el programa llega a la línea 300. En tal caso la puntuación obtenida se incrementa con el valor del número recién comido. Se presenta la cabeza sobre la pantalla y aparece la puntuación alcanzada. El bucle FOR ... NEXT que se extiende entre las líneas 310 y 360 añade a la serpiente el número adecuado de segmentos. Este número está determinado por el valor del número que la serpiente acaba de engullir. Cada vez que se recorre el bucle, la línea 320 llama a la rutina de INKEY$; la línea 325 se ocupa de comprobar si la cabeza sigue estando todavía en una situación permitida. La línea 330 incrementa el puntero de cabeza y la 340 almacena la nueva posición de la cabeza en la matriz s. La cabeza se presenta en la pantalla por medio de la línea 350. Para añadir los segmentos adicionales, lo que se hace es no suprimir la cola como haríamos normalmente. El hecho de no borrar la cola significa que verías como la serpiente iba corriendo cada vez más si no se pusiera el retardo de la línea 355.

La última sección del programa, que es la que empieza en la línea 2000, es la rutina de «Final de Juego». La línea 2000 hace que aparezca el mensaje FIN! ocupando toda la pantalla, y además hace que suenen una serie de BEEPs. En la línea 2010 se actualiza el valor de la puntuación máxima alcanzada, en el supuesto de que supere al valor anterior de dicha puntuación máxima; la línea 2020 se ocupa de presentar la puntuación actual y la máxima. Por último, las líneas 2030 y 2040 permiten al jugador que vuelva a empezar.



La serpiente engulle los números que encuentra a su paso. Pero no puede retroceder sobre sí misma. La dificultad aumenta a medida que su longitud crece.







¿TE GUSTARIA SER UN HEROE?
VIAJA EN EL TIEMPO POR
ESTA MULTIAVENTURA GRAFICA
PARA ENCONTRAR Y DESTRUIR
EL ARMA SECRETA ALEMANA.
AMSTRAD
cassette – 2000 pts
disco – 4000 pts
SPECTRUM 2000 pts
COMMODORE 2000 pts
Biggles
EN TODOS LOS CINES DE ESPAÑA
DISTRIBUIDA POR C.B. FILMS
¡VIVE LA PELICULA DESDE TU ORDENADOR!
Proximamente a la venta en
la nueva Konami shop
C/ Francisco Navacerrada 19
MIRRORSOFT
SERMA
Pídelo a SERMA C/. CARDENAL BELLUGA, 21. 28028 MADRID. Telfs. 256 50 06/05/04/03

EL PROCESAMIENTO DE TEXTOS

Si quieres poner a punto tu habilidad mecanográfica a expensas de un procesador de textos hecho y derecho, la respuesta la tienes en este editor de textos de fácil programación.

Ya conoces las ventajas del procesado de textos. Realmente ésta es una de las aplicaciones más útiles de los ordenadores domésticos. Para conseguir los mejores resultados, puedes comprarte el *software* existente en código máquina, pero normalmente está diseñado para usos profesionales y puede resultar muy caro.

La parte de programa que ofrecemos en este artículo presenta un buen editor de textos para quien da sus primeros pasos en este terreno, permitiéndole adquirir una sensación real de cómo son los programas de este tipo.

Con este programa puedes crear informes, cartas o cualquier otra forma de correspondencia. Naturalmente la salida se hace a través de la impresora. El texto puede ser almacenado en cinta o **Microdrive** y puede recuperarse para su posterior utilización, ya sea por este programa o cualquier otro que haga uso de los ficheros secuenciales creados con los datos.

Debido a la longitud del programa completo —no menos de 7,5K— el programa se ha fraccionado en tres unidades de fácil digestión, y no funcionará adecuadamente hasta que se haya introducido la última parte. Sin embargo, después de introducir la segunda parte ya estarás en condiciones de crear ficheros de texto. La tercera parte contiene la rutina de impresora, que evidentemente necesitarás para obtener copias de tus escritos. También se incluyen unas facilidades de ordenación y búsqueda, así como una opción para hacer modelos de cartas.

Antes de introducir la primera parte del programa, echemos una ojeada a la forma en que se utiliza. Se ha configurado de una forma que resulte de uso cómodo en la mayoría de las aplicaciones domésticas, pero como ocurre con todo programa de este tipo puedes modificar el contenido de los diversos menús y mensajes de pantalla para que se adapten a tus necesidades y preferencias. No obstante, si decides realizar dicha modificación, deberás respetar las mismas convenciones de numeración de líneas para que el programa se enlace correctamente con otros módulos, especialmente cuando se le añadan otros.

EL MENU PRINCIPAL

En cuanto ejecutes (con RUN) la versión completa de la copia del programa te aparecerá el menú general, que te ofrece:

1. CARGAR TEXTO.
2. GUARDAR TEXTO.
3. CAMBIAR COLOR.
4. EDITAR.
5. BORRAR TEXTO.
6. IMPRIMIR TEXTO.
7. MODIFICAR PARAMETROS.
8. SALIR DEL PROGRAMA.

LOAD

Si quieres seguir trabajando sobre un fichero de texto ya existente, que anteriormente habías guardado (con el comando SAVE), pulsa 1 para arrancar la rutina de carga. Ten en cuenta que el texto cargado se escribirá sobre cualquier otro que tengas en memoria. Seguidamente te pedirá el nombre de un fichero y tendrás que proporcionárselo para que el proceso continúe. El fichero de texto cuyo nombre hayas tecleado, será cargado en la memoria, pudiendo trabajar sobre él a tu antojo.



SAVE

La rutina SAVE se utiliza para la creación de ficheros secuenciales para almacenamiento de datos. Cuando selecciones esta opción pulsando la tecla S, lo primero que te pedirá el programa es un nombre de fichero. El **Spectrum** almacena los datos independientemente de que tengas o no algún texto en la memoria. No se aceptará una entrada nula. El sistema ya «sabe» el dispositivo que estás utilizando y continúa enviándote mensajes en consecuencia, acerca de la preparación de la cinta o la unidad de disco. Por último, pulsa [ENTER] para que se efectúe la operacion de almacenamiento.

Se pueden añadir otras rutinas adicionales de SAVE que permitan la protección de los ficheros, el cambio



- NOMBRES DE FICHEROS
- EDICION
- PARAMETROS DE ENTRADA Y SALIDA
- MANIPULACION DEL TEXTO
- PROGRAMA EDITOR

de nombre o su sustitución. En el listado que presentaremos en nuestro próximo artículo podrás examinar varias rutinas de muestra.

ENTRADA/SALIDA

El programa es válido para cinta o **Microdrive**. A continuación te damos las líneas que deberás cambiar en el listado principal si decides utilizar el **Microdrive.**

```
6005 CAT 1
6010 INPUT "NOMBRE DEL
     FICHERO",LINE N$:IF LEN
     N$<1 OR LEN N$>10 THEN
     GO TO 6010
6015 LOAD *"M",1;N$ DATA
     T$()
6205 CAT 1
6230 SAVE *"M",1;N$ DATA T$
     ():GO TO 6020
```

MODO DE EDICION

Si seleccionas E en el menú principal, entras en el modo de edición y te aparecerá un menú secundario:

1. PRINCIPIO DE TEXTO.
2. FIN DE TEXTO.
3. LINEA SIGUIENTE.
4. VOLVER A MENU PRINCIPAL.

Puedes seleccionar una de las primeras opciones para trasladarte al principio o al final del texto que tienes en memoria. Si utilizas la tercera opción, puedes volver a la última línea editada o salir al menú principal.

Cada una de las tres primeras opciones te sitúa en un modo de entrada. La pantalla presenta las líneas del principio o el final del texto (sólo puede presentar a un tiempo una de las dos cosas si ya tienes algo en memoria). Cerca de la parte inferior de la pantalla hay una «zona de trabajo» separada.

El texto introducido en la zona inferior se transfiere a la memoria y a la zona de pantalla cuando se pulsa la tecla ENTER, o de forma automática cuando se han completado dos líneas de pantalla.

El programa lleva incorporadas varias utilidades de edición. Toda la labor de edición e introducción tiene lugar en la zona de trabajo.

Los controles de edición te permiten desplazarte hacia adelante y atrás a lo largo de la línea de texto del área de trabajo para insertar o suprimir caracteres antes de que el texto sea transferido a la memoria.

En otras palabras, el texto que ya está en memoria, representado en la parte alta de la pantalla, tiene que ser copiado línea a línea a la zona de trabajo para poder trabajar sobre él.

Puedes introducir nuevas líneas en cualquier parte del cuerpo principal del texto. Para hacerlo, pulsa la tecla adecuada del modo «editor» (consulta las instrucciones detalladas) y coloca la marca debajo del punto donde quieras insertar la nueva línea.

También puedes suprimir texto línea a línea entrando en el modo editor y colocando nuevamente la marca debajo de la línea que quieras borrar antes de pulsar la tecla adecuada de borrado.

Para volver al menú del modo editor, pulsa CAPS SHIFT + SYMBOL-SHIF adecuada y desde allí al menú principal para modificar cualquiera de los parámetros del sistema.

COLOR Y BORRADO

Con la opción COLOR puedes ajustar a tu gusto el color de la pantalla.

Otra de las funciones disponibles en el menú principal es el borrado de la memoria, pero sólo después de que hayas respondido «sí» al mensaje «Estás seguro?» que te aparecerá en pantalla en cuanto pulses la tecla 5. Ten en cuenta que esta opción no inicializa los parámetros de entrada/salida ni la impresora. Si quieres arrancar de nuevo desde cero, pulsa 8 para salir totalmente del programa, con lo que se produce una inicialización completa del sistema.

Teclea para Spectrum

```
1 REM PROCESADOR TEXTOS 1.
  PARTE
10 POKE 23659,3
20 BORDER 7: PAPER 7: INK 9:
   CLS
30 DIM L(8): FOR N=1 TO 8:
   READ L(N): NEXT N
40 LET Z$=""
100 LET EXT=200: DIM T$(EXT,
    32): LET LL=32: LET PL=
    32
105 LET S$="[32*ESPACIO]"
110 LET T$(1)="INICIO DEL
    FICHERO DE TEXTO": LET
    T$(2)="-----------------
    ---------------": LET T$
    (3)=S$
120 LET T$(4)=S$: LET T$(5)=
    "----------------------
    ---------": LET T$(6)="T
    TFIN DEL FICHERO DE
    TEXTO"
130 LET T=1: LET B=6: LET P=4
```



```
140 CLS : PRINT  INVERSE 1;AT
    0,10;" MENU PRINCIPAL "
150 PRINT AT 4,8;"1:-CARGAR
    TEXTO";AT 6,8;"2:-GUARDAR
    TEXTO";AT 8,8;"3:-CAMBIAR
    COLOR";AT 10,8;"4:-EDITAR
    ";AT 12,8;"5:-BORRAR
    TEXTO";AT 14,8;"6:-
    IMPRIMIR TEXTO";AT 16,8;
    "7:-MODIFICAR PARAMETROS
    ";AT 18,8;"8:-SALIR DEL
    PROGRAMA"
160 PRINT #1;TAB 7;"ELIGE
    OPCION (1-8)"
170 LET A$=INKEY$: IF A$=""
    THEN  GO TO 170
180 IF A$<"1" OR A$>"8" THEN
    GO TO 170
190 LET A=VAL A$: CLS
200 GO SUB L(A)
210 GO TO 140
500 CLS : PRINT  INVERSE 1;
    AT 4,10;" MENU DEL
    EDITOR "
510 PRINT AT 8,8;"1:
    -PRINCIPIO DE TEXTO";AT
    10,8;"2:-FIN DE TEXTO";
    AT 12,8;"3:-LINEA
SIGUIENTE";AT 14,8;"4:
    -VOLVER MENU PRINCIPAL"
520 PRINT AT 18,7;"ELIGE
    OPCION (1-4)"
530 LET A$=INKEY$: IF A$=""
    THEN  GO TO 530
540 IF A$<"1" OR A$>"4" THEN
    GO TO 530
550 LET A=VAL A$: CLS
560 IF A=4 THEN  RETURN
570 IF A=1 THEN  LET P=4
580 IF A=2 THEN  LET P=B-2
590 GO SUB 1000: GO SUB 2000
600 GO TO 500
900 PRINT AT 10,8;"ESTAS
    SEGURO?": PAUSE 0
910 IF INKEY$="S" THEN  RUN
920 RETURN
4000 RETURN
6000 REM CARGA
6010 INPUT "NOMBRE DEL
     FICHERO", LINE N$: LOAD
     N$ DATA T$()
6020 LET B=VAL T$(1): LET T$
     (1)="INICIO DEL FICHERO
     DE TEXTO": RETURN
6200 REM GUARDAR
6210 LET T$(1)=STR$ B
6220 INPUT "NOMBRE DEL
     FICHERO", LINE N$: IF
     N$="" OR LEN N$>10 THEN
     GO TO 6220
6230 SAVE N$ DATA T$(): GO
     TO 6020
6500 INPUT AT 0,0;"ANCHURA
     DE PANTALLA (1-80)",PL:
     IF PL<1 OR PL>80 THEN
     GO TO 6500
6510 INPUT AT 0,0;"NUMERO DE
     CARACTERES POR LINEA
     (1-";(PL);")";LL: IF
     LL<1 OR LL>PL THEN  GO
     TO 6510
6520 LET LL=LL+1: RETURN
9000 DATA 6000,6200,3000,500
     ,900,4000,6500,9999
```

EL ZOCO DE INPUT



Todo se compra y se vende. Los antiguos zocos fueron lugares destinados a todo tipo de transacciones. INPUT también tiene el suyo. Vuestras operaciones de compra, cambio o venta serán publicadas en esta sección, pero dos son las limitaciones que imponemos:

a) La propuesta tendrá que ver con la microinformática.

b) Nos reservamos el derecho de no publicar aquellos insertos de los que se sospeche un trasfondo lucrativo.

Ahora un ruego. Tratar de resumir al máximo el texto; escribir casi como un telegrama siendo claros y concisos.

Envía tu mensaje a:

INPUT MSX ZOCO
P.º de la Castellana, 93. Planta 14
28046 MADRID

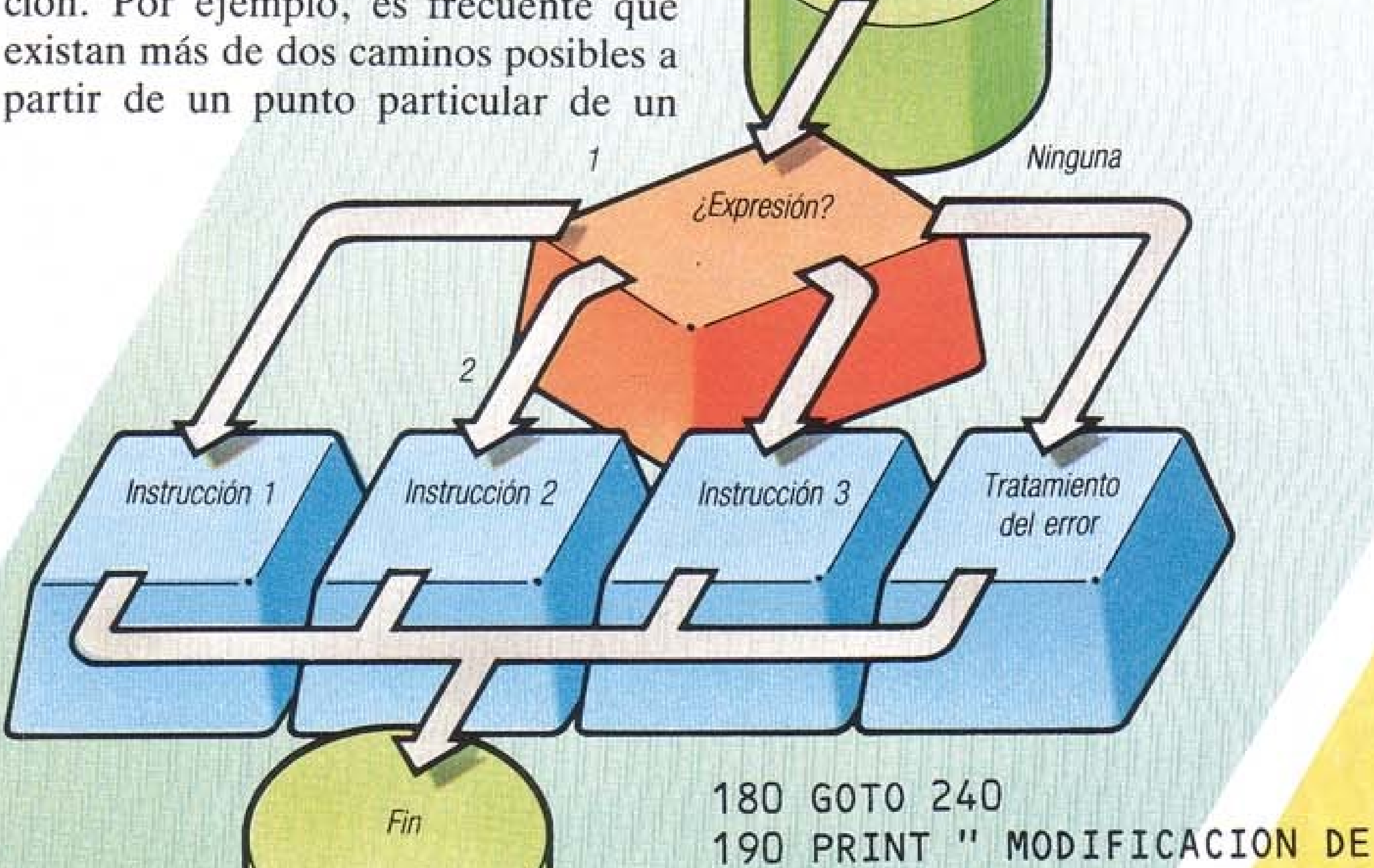
ESTRUCTURA TUS PROGRAMAS (II)

Este artículo constituye una segunda parte de aquel otro mismo título publicado en nuestro número 5. Aquí continuamos repasando algunas estructuras fundamentales en programación.

En general las estructuras IF ... THEN y WHILE ... DO son suficientes para casi todos los programas. Existen sin embargo otras estructuras que simplifican aún más la programación. Por ejemplo, es frecuente que existan más de dos caminos posibles a partir de un punto particular de un programa. Se podrían utilizar en tal caso varias estructuras IF ... THEN anidadas, pero resulta más cómodo usar una estructura de elecciones múltiples.

El diagrama de bloques de esta estructura, llamada CASE, se muestra en la figura 1, y en BASIC se escribe así:

```
100 REM BLOQUE DE OPCIONES
110 IF C$="C" THEN GOTO 170
120 IF C$="V" THEN GOTO 190
130 IF C$="A" THEN GOTO 210
140 IF C$="E" THEN GOTO 230
150 PRINT "OPCION NO
    RECONOCIDA"
160 GOTO 240
170 PRINT "CREACION DE UN
    REGISTRO": GO SUB 1000
```



1. Opciones múltiples

```
180 GOTO 240
190 PRINT " MODIFICACION DE
    UN REGISTRO":GOSUB 2000
200 GOTO 240
210 PRINT "ANULACION
    DE UN REGISTRO":
    GOSUB 3000
220 GOTO 240
230 PRINT "FIN DEL
    BLOQUE"
```

Con las instrucciones ON ... GOTO y ON ... GOSUB se puede disponer de un método aún más eficaz para realizar las elecciones múltiples. Por medio de la sentencia ON ... GOTO puedes asegurarte de que todas las opciones del programa contienen el GOTO adecuado para salir con normalidad de la propia opción, como puedes ver en el ejemplo siguiente: después de cada opción hay una sentencia GOTO 1210 para reconducir el programa al final de la rutina.

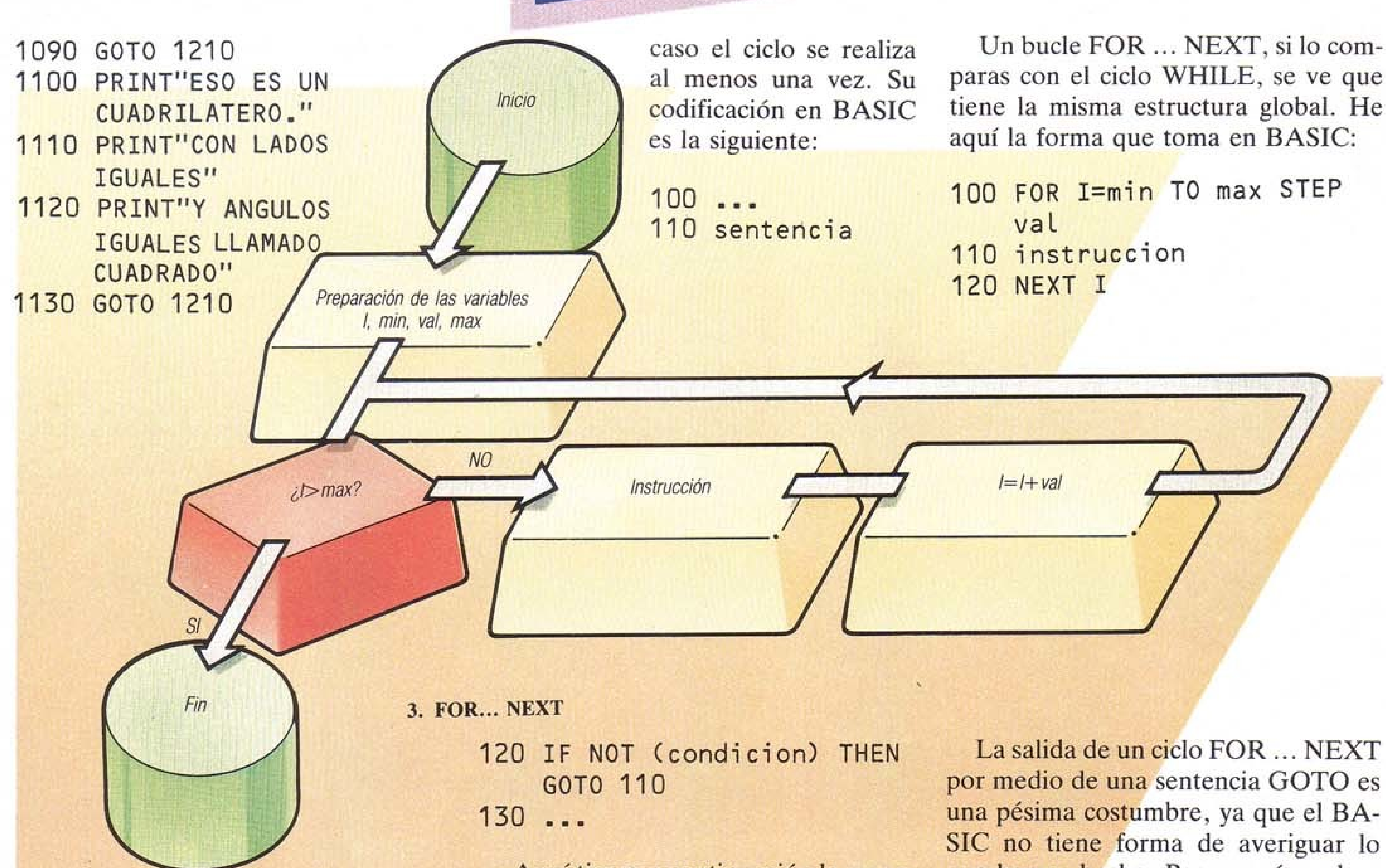
```
1000 SUBRUTINA PARA UN
     POLIGONO
1010 INPUT "CUANTOS LADOS
     DESEAS";N
1020 IF N=3 THEN GOTO 1060
1021 IF N=4 THEN GOTO 1100
1022 IF N=5 THEN GOTO 1140
1023 IF N=6 THEN GOTO 1160
1024 IF N=7 THEN GOTO 1180
1025 IF N=8 THEN GOTO 1200
1030 PRINT "NO CONOZCO
     NINGUN POLIGONO"
1040 PRINT "DE ";N;" LADOS"
```

Inicio
Instrucción
¿Condición?
FALSA
VERDADERA
Fin

2. Ciclo REPEAT... UNTIL

```
1050 GOTO 1210
1060 PRINT"ESO ES UN
     TRIANGULO."
1070 PRINT"CON LADOS
     IGUALES."
1080 PRINT"DENOMINADO
     EQUILATERO."
```

```
1090 GOTO 1210
1100 PRINT"ESO ES UN
     CUADRILATERO."
1110 PRINT"CON LADOS
     IGUALES"
1120 PRINT"Y ANGULOS
     IGUALES LLAMADO
     CUADRADO"
1130 GOTO 1210
```



3. FOR... NEXT

```
1140 PRINT"ESO ES UN
     PENTAGONO"
1150 GOTO 1210
1160 PRINT"ESO ES UN EXAGONO"
1170 GOTO 1210
1180 PRINT"ESO ES UN
     HEPTAGONO"
1190 GOTO 1210
1200 PRINT"ESO ES UN
     OCTOGONO"
1210 PRINT
1220 RETURN
```

Como se trata de una subrutina, tiene que haber un programa desde el que se la llame, del que nos ocupamos en el siguiente apartado.

REPEAT ... UNTIL

La estructura REPEAT ... UNTIL constituye otra forma de crear un bucle dentro de un programa. También en este caso es posible simular esta estructura cuando no exista en la versión de BASIC utilizada. A diferencia de la estructura WHILE ... DO, en este caso el ciclo se realiza al menos una vez. Su codificación en BASIC es la siguiente:

```
100 ...
110 sentencia
120 IF NOT (condicion) THEN
    GOTO 110
130 ...
```

Aquí tienes a continuación la manera de escribir un programa con un ciclo REPEAT, utilizando la subrutina del ejemplo anterior:

```
10 PRINT"PUEDO DECIRTE LOS
   NOMBRES"
20 PRINT"DE ALGUNOS
   POLIGONOS"
30 REM INICIO DEL BUCLE
40 GOSUB 1000
50 INPUT"DESEAS OTRO NOMBRE
   (S/N)";A$
60 IF A$="S" THEN GOTO 30
70 PRINT"ADIOS!!"
```

La línea 1000 es la subrutina de los polígonos que hemos visto hace poco.

CICLOS FOR ... NEXT

El familiar bucle FOR ... NEXT, en realidad no es más que un caso particular del ciclo WHILE ... DO: en efecto, se utiliza cuando se conoce con antelación el número de veces que recorrerá el ciclo, por lo que se especifica al principio. La variable que va contando el número de pasadas se llama variable de control.

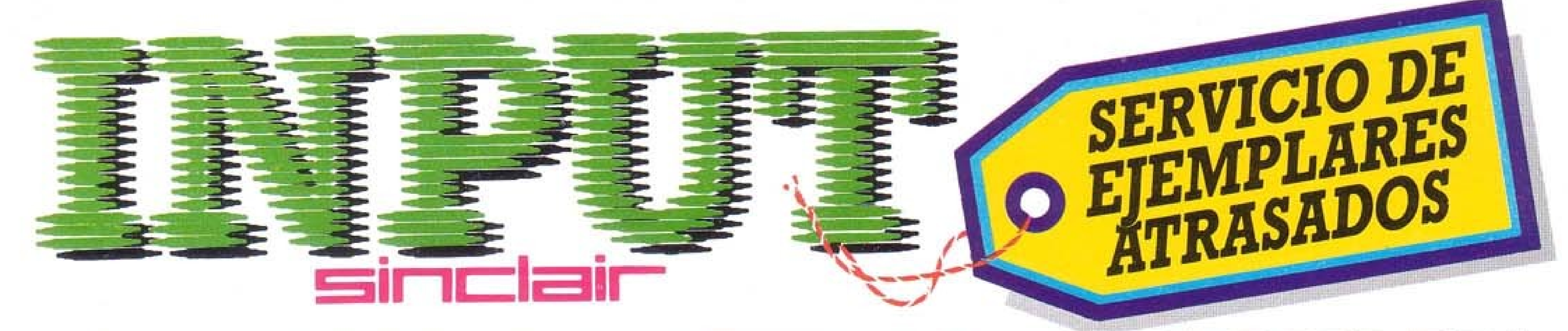
Un bucle FOR ... NEXT, si lo comparas con el ciclo WHILE, se ve que tiene la misma estructura global. He aquí la forma que toma en BASIC:

```
100 FOR I=min TO max STEP
    val
110 instruccion
120 NEXT I
```

La salida de un ciclo FOR ... NEXT por medio de una sentencia GOTO es una pésima costumbre, ya que el BASIC no tiene forma de averiguar lo que hemos hecho. Peor es aún volver a entrar a mitad del ciclo después de haber salido de él: un programa de este tipo sería muy difícil de descifrar. Debes tratar por todos los medios de evitar semejantes pasos peligrosos.

PONGAMOSLO TODO JUNTO

Las estructuras presentadas son suficientes para cualquier programa que pretendas escribir. Pero hasta que se completa un programa, el camino que resta es todavía largo. Hay que especificar las variables que se vayan a utilizar y comprobar que las variables usadas en módulos distintos no presenten contradicciones entre ellas. Es necesario establecer además cuál será la entrada del programa y cuál será su salida. Por último, hay que probar cada módulo individual antes de enlazarlo con los demás. Puedes echar una atenta ojeada a algún programa para ver si reconoces su estructura. También puedes probar a mejorar un programa, haciéndolo más estructurado; puedes elegir entre tus propios programas o los de las revistas.



¡NO TE PIERDAS NI UN SOLO EJEMPLAR!

INPUT SINCLAIR quiere proporcionar a sus lectores este nuevo servicio de ejemplares atrasados para que no pierdan la oportunidad de tener en sus hogares todos los ejemplares de esta revista, líder en el mercado español.

Podréis solicitar cualquier número de **INPUT SINCLAIR** que querais, siempre al precio de cubierta (sin más gastos).

Utiliza el cupón adjunto, enviándolo a **EDISA** (Dpto. de Suscripciones), López de Hoyos, 141 - 28002 Madrid, o bien llámanos por teléfono al (91) 415 97 12.





INPUT sinclair
13
MAS ENTORNO A LAS INTERRUPCIONES
AGREGALE INSTRUCCIONES AL BASIC
PUZZLES Y MATEMATICAS

siempre a tu servicio

CUPON DE PEDIDO

SI, envíenme contrarreembolso .... ejemplares de **INPUT SINCLAIR** de los números:

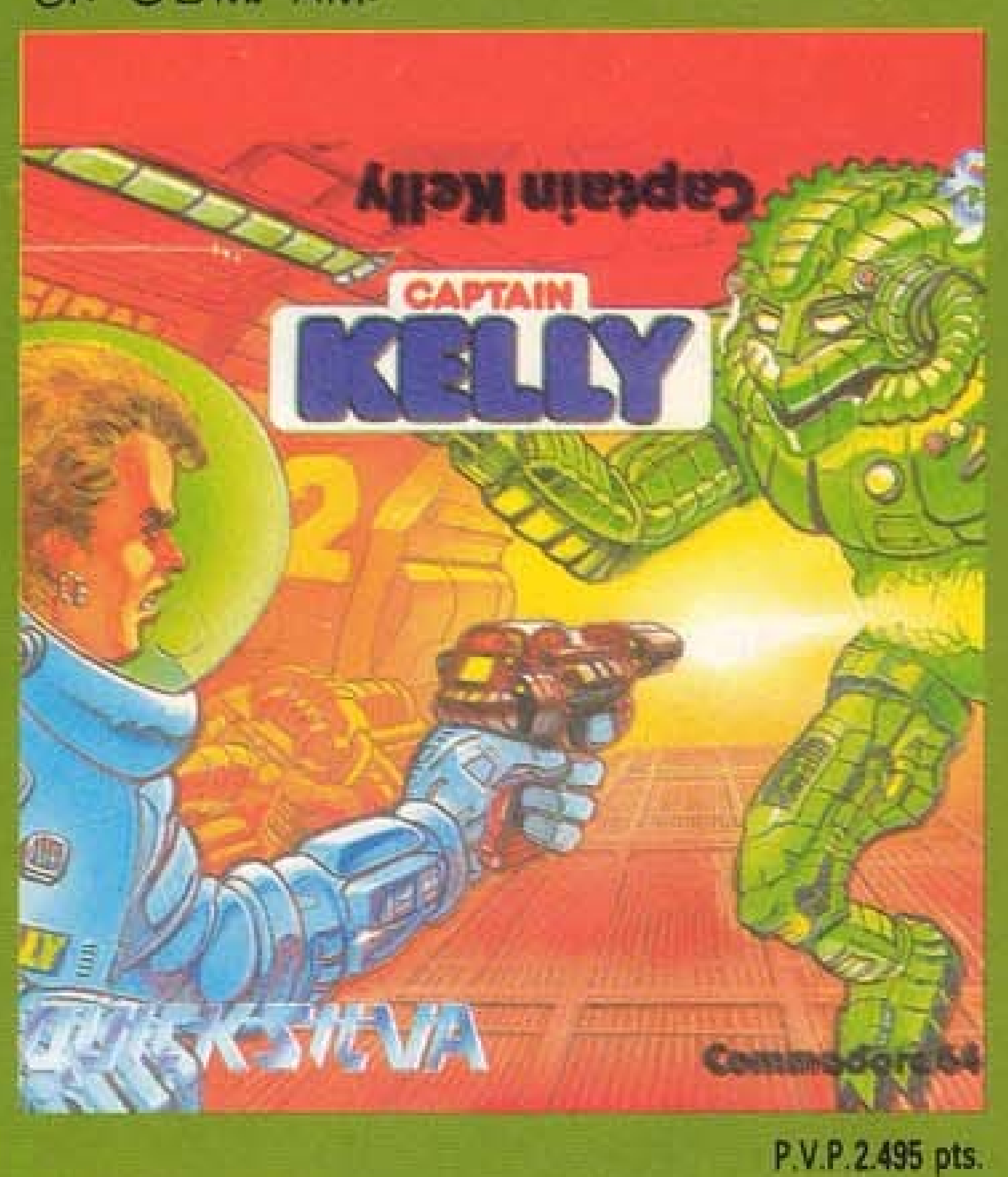
(marca con una (X) tu elección)

[1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13]

NOMBRE
APELLIDOS
DOMICILIO
NUM. PISO ESCALERA COD. POSTAL
POBLACION PROV.
TELEFONO FIRMA

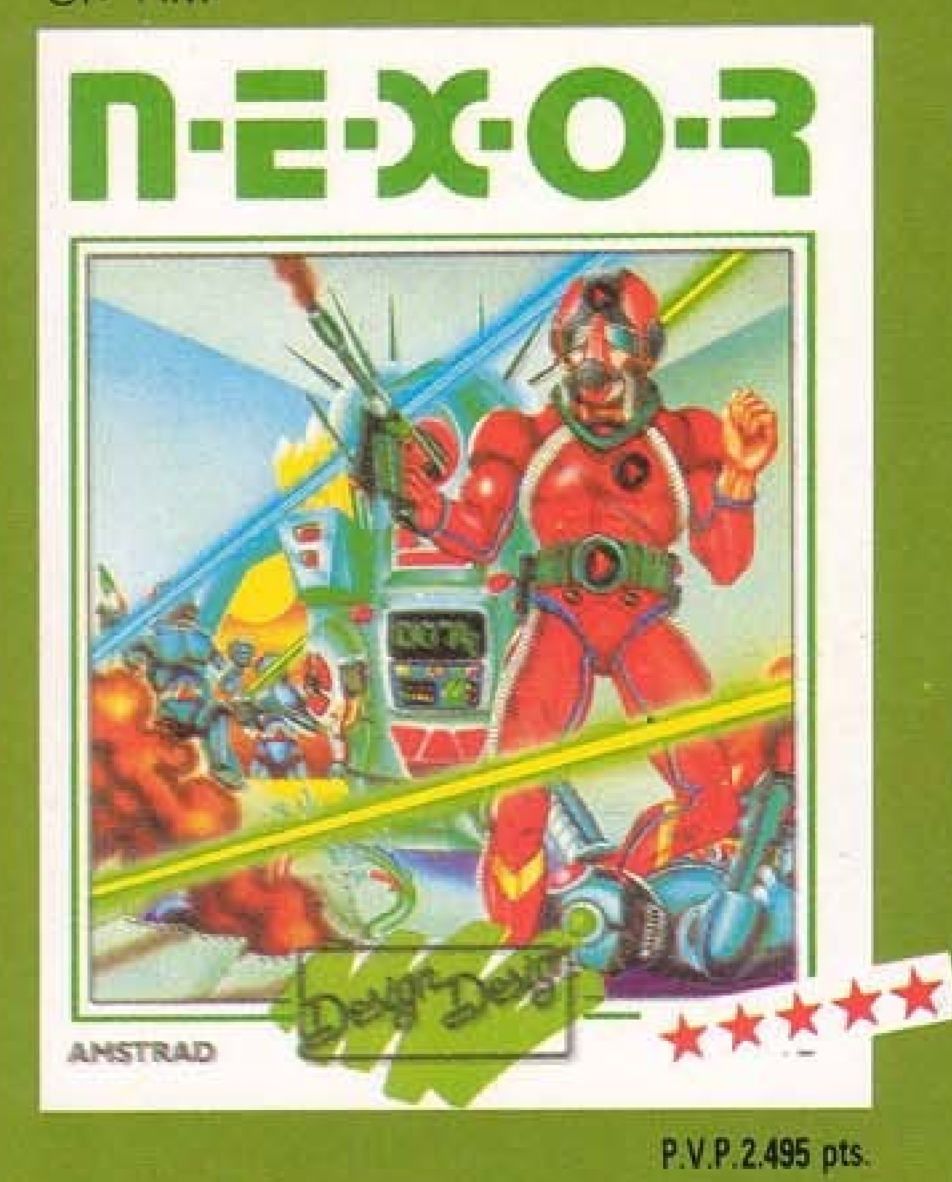
MIND GAMES ESPAÑA, S.A.

SP. CBM. AM.



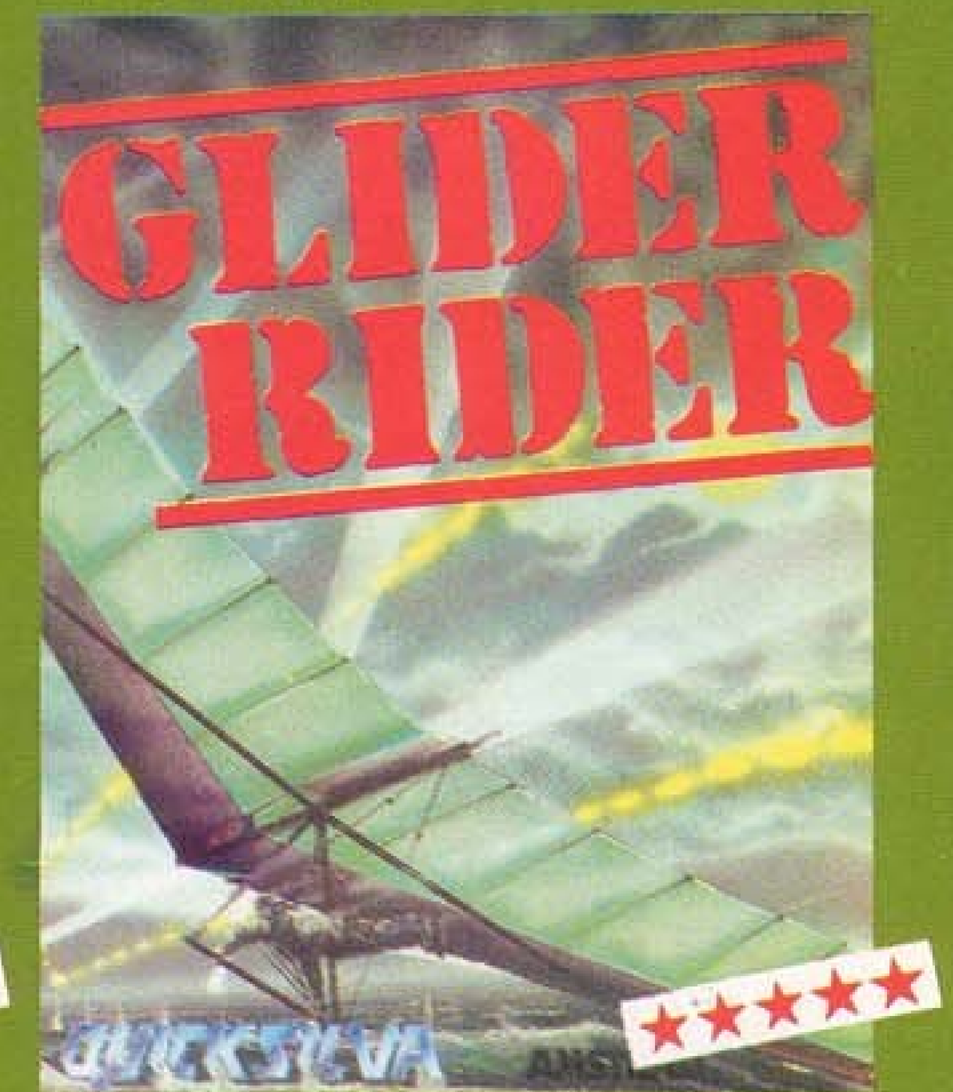
P.V.P. 2.495 pts.

SP. AM.



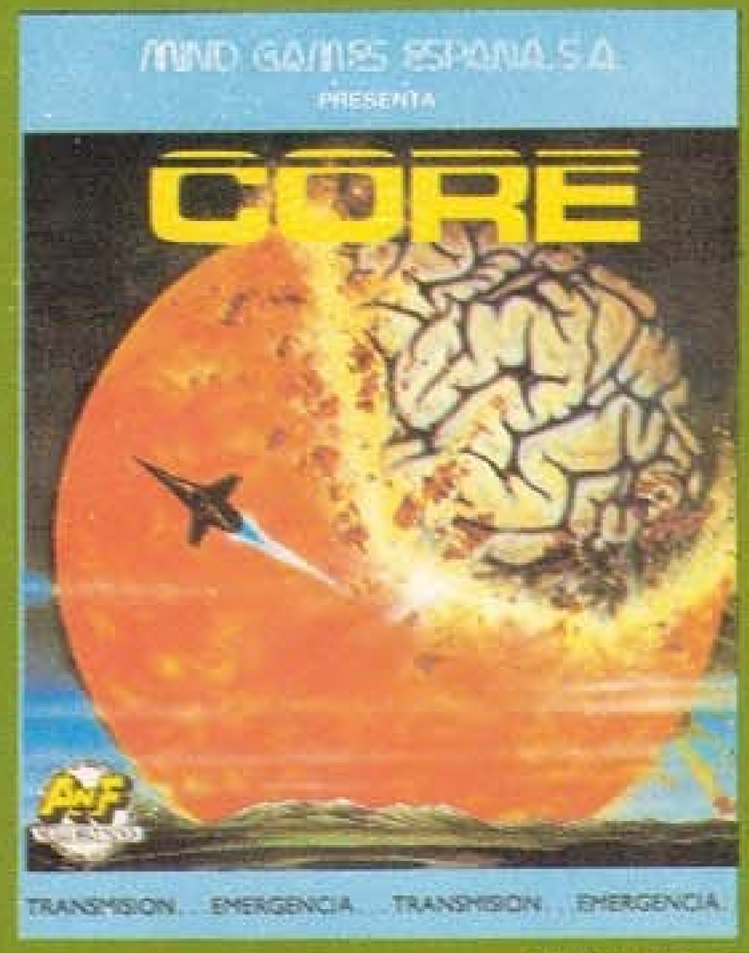
P.V.P. 2.495 pts.

SP. CBM. AM.



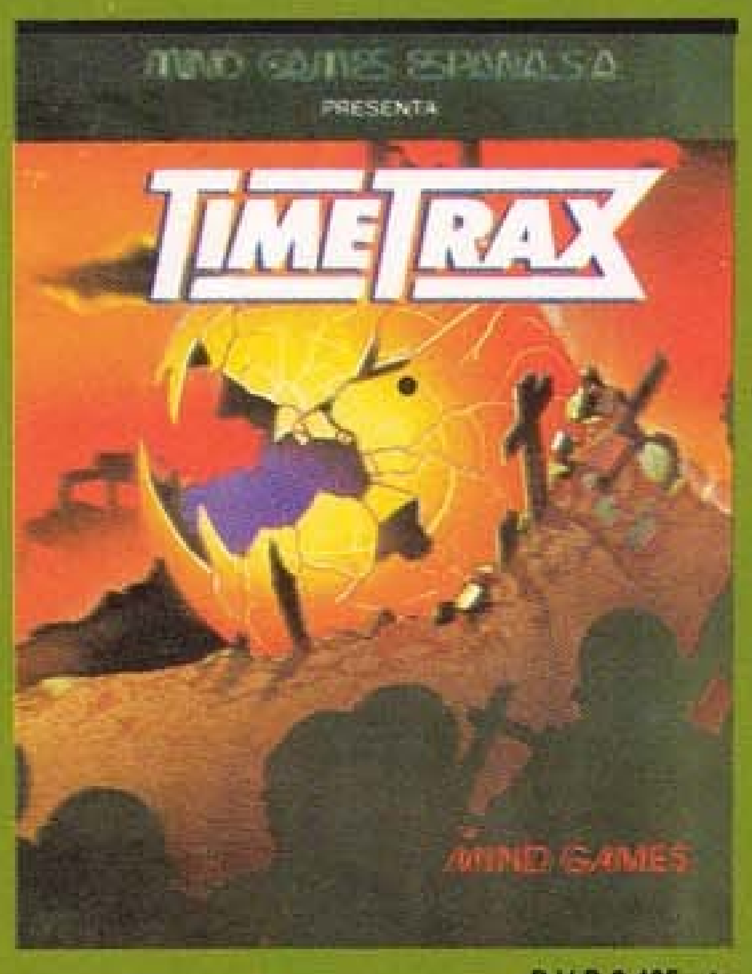
P.V.P. 2.495 pts.

SP. CBM. AM.



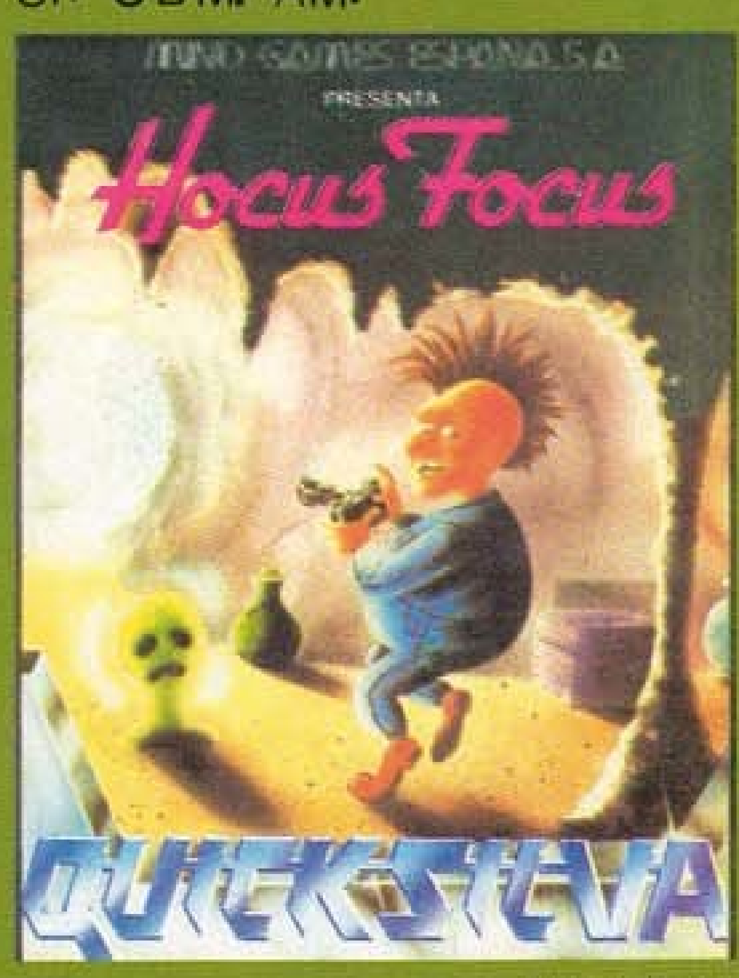
P.V.P. 2.495 pts.

SP. CBM. AM.



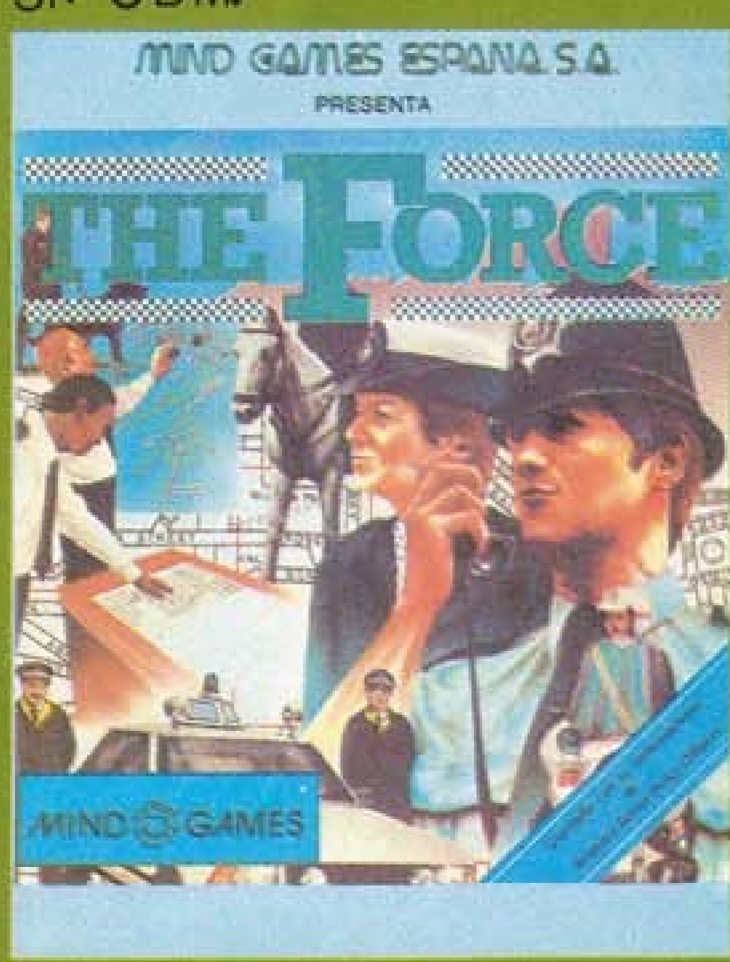
P.V.P. 2.495 pts.

SP. CBM. AM.



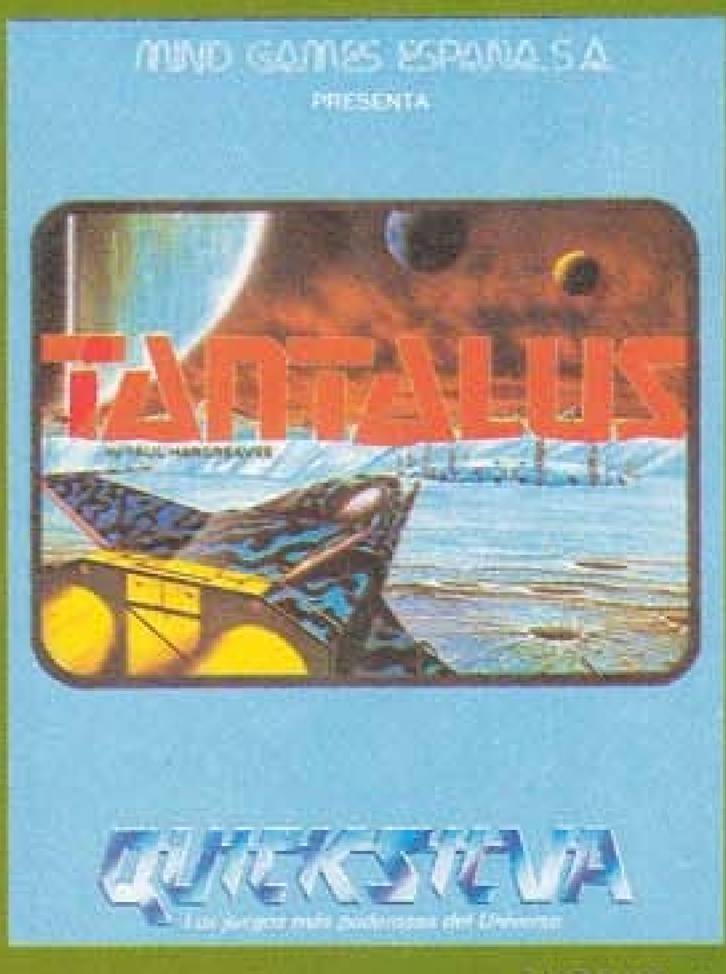
P.V.P. 2.495 pts.

SP. CBM.



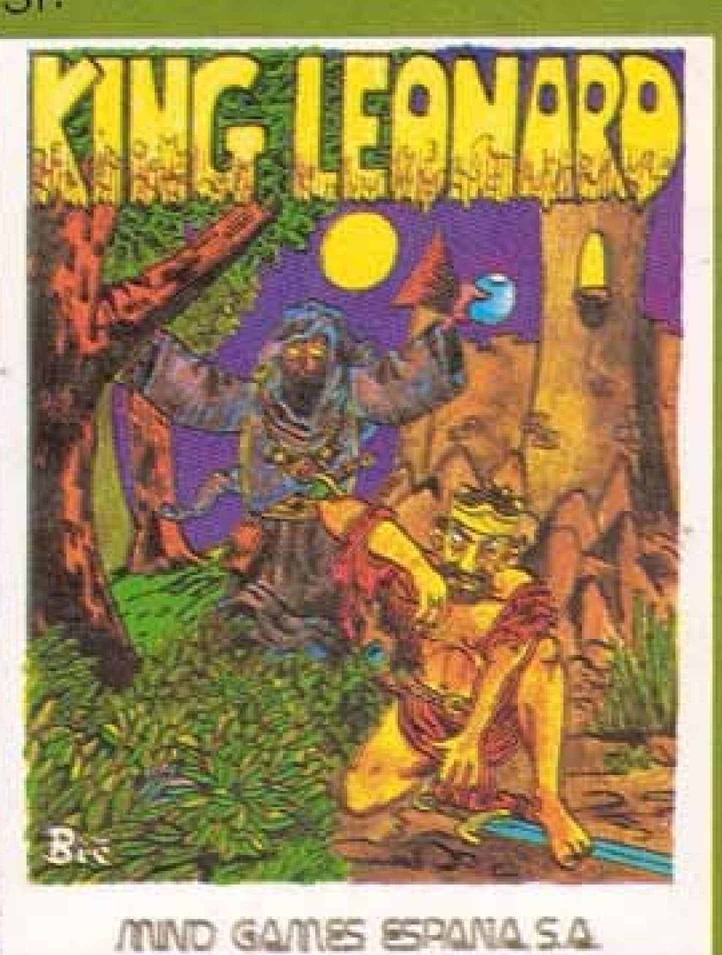
P.V.P. 2.495 pts.

SP.



P.V.P. 2.495 pts.

SP.

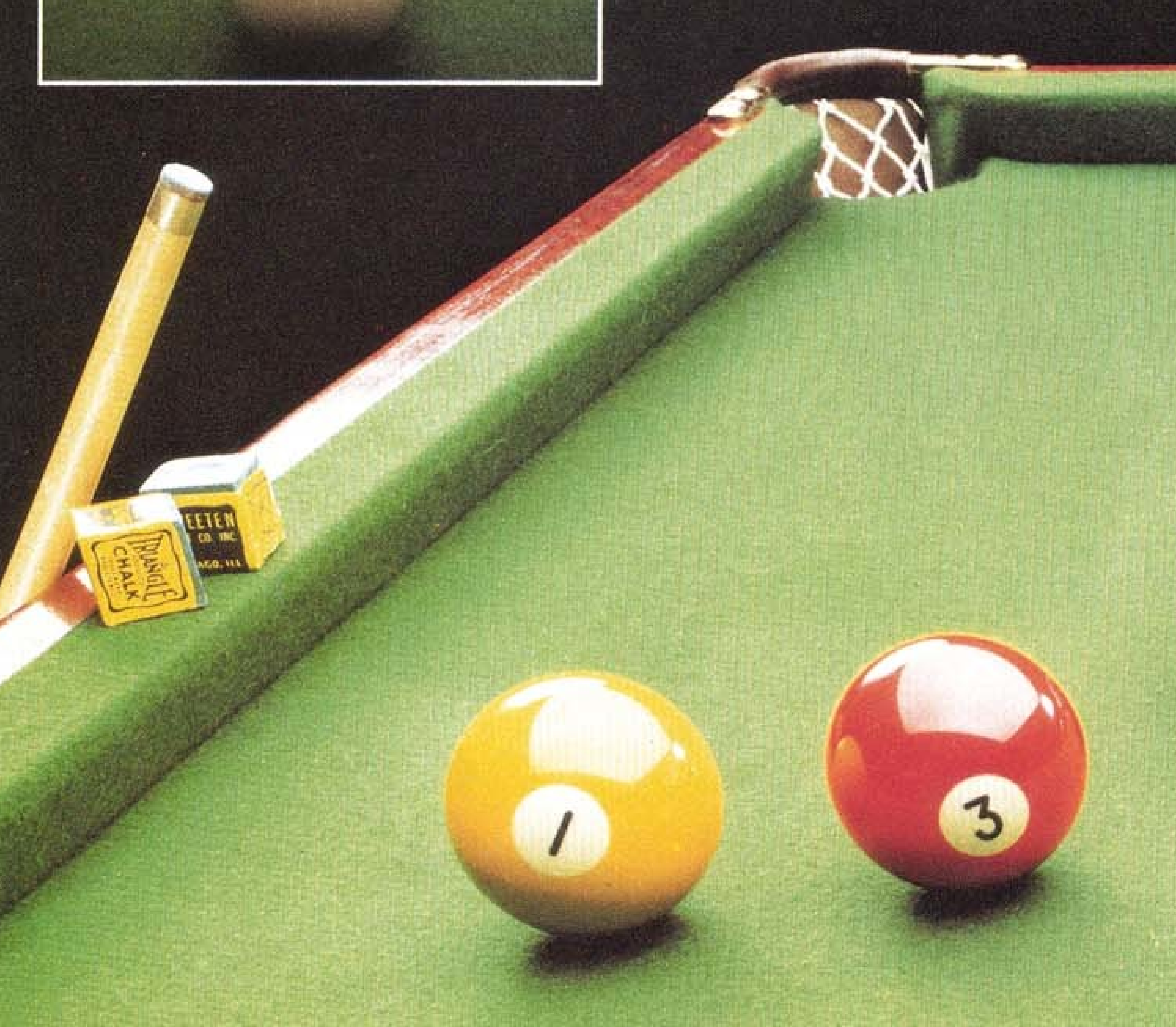


P.V.P. 2.495 pts.

Precio Venta Público recomendado: — I.V.A. incluido. Disponibles en Galerías Preciados y en todos los establecimientos de Software, o directamente a MIND GAMES ESPAÑA, S.A.: Mariano Cubí, 4, entlo. 1.ª - 08006-BARCELONA - Tel. (93) 218 34 00



PON TUS PROGRAMAS EN ORDEN

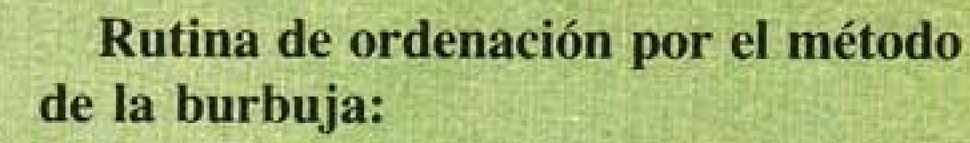


Cuando hayas terminado con el diseño general del programa y escrito todos los módulos individuales, puedes empezar a pensar en la forma de probar los módulos. Después tendrás que ocuparte de la forma en que vas a combinar todos los elementos.

La mayoría de las subrutinas y módulos necesitarán unirse con el resto del programa de una u otra forma. Esto se hace normalmente utilizando variables. Algunas variables, que se especifican al principio, se pasarán a la rutina constituyendo lo que se llama parámetros de entrada. Hay otras variables que serán introducidas en el programa por la propia rutina, son los parámetros de salida. Es muy importante que las variables se especifiquen de forma precisa y que no puedan ser confundidas.

Cuando empieces a escribir un programa, debes hacer una lista de las variables que necesites usar, junto con una descripción de su uso y sus posibles valores, si los conoces. De no hacerlo así, aunque puede ser que al principio del programa sepas muy bien lo que significa cada letra, es más que probable que se te haya olvidado cuando vuelvas sobre el programa más adelante. Resulta de gran ayuda la utilización de nombres largos de variables, en los casos en que tu ordenador lo permite, a menos que andes muy escaso de espacio en la memoria.

Aquí tienes un ejemplo de la forma en que podrías especificar las variables en una rutina de ordenación por el método de la burbuja:



Rutina de ordenación por el método de la burbuja:

Ordenación de la parte especificada de una matriz según el orden alfabético.

Variables de entrada:

A$(N) matriz unidimensional que se quiere ordenar (valor de N≥1)

N1 primer elemento de la matriz que se quiere ordenar (1≤N1≤N2)

N2 último elemento de la matriz que se quiere ordenar (N1≤N2≤ longitud de A)

Variables de salida:
A$(N) matriz ordenada

Variables temporales:
Z, Z$, I

Es útil anotar las variables temporales utilizadas dentro de una subrutina para evitar conflictos entre los mó-

Avanza un paso más en la estructuración de tus programas, tomando como ejemplo la construcción de un programa de ordenación por el método de la burbuja, con el que podrás ordenar de todo.

- SEGUIMIENTO DE LAS VARIABLES DEL PROGRAMA
- SUBRUTINAS
- COMO PROBAR LOS MODULOS
- PONIENDOLO TODO JUNTO



dulos o con el resto del programa. También te será de utilidad reservar algunas letras especialmente para dichas variables temporales. Por ejemplo, podrías usar de Z0 a Z9. Con esto además consigues evitar el derroche de espacio para las variables.

Si organizas bien todos estos aspectos al principio de la construcción de un programa, te ahorrarás muchos quebraderos de cabeza más adelante. Muchos errores de programación o «*bugs*» se deben a que las variables han sido corrompidas, es decir, han cambiado de valor cuando tú no esperabas que lo hicieran. Pero si sigues el método anterior, no tendrás ningún problema.

La redacción de una lista de variables también te será útil si alguna vez cambias el programa más adelante, ya que entonces será el momento de consultar la lista donde se especifica el modo en que se utilizan dichas variables. Con esa lista te resultará mucho más rápida la modificación de las variables y te evitarás la introducción de nuevos errores en la programación, debidos a la corrupción de otras variables que ya habían sido usadas para otros fines. Acuérdate de anotar las modificaciones que introduzcas, así como la fecha de las mismas.

Aquí tienes ya el listado de la rutina de ordenación por el método de la burbuja.

```
1000 REM METODO DE LA BURBUJA
        (A$(N),N1,N2)
1010 LET Z=0
1020 FOR I=N1 TO N2-1
1030 IF A$(I)<=A$(I+1) THEN
     GOTO 1080
1040 LET Z$=A$(I)
1050 LET A$(I)=A$(I+1)
1060 LET A$(I+1)=Z$
1070 LET Z=1
1080 NEXT I
1090 IF Z=1 THEN GOTO 1010
1100 RETURN
```

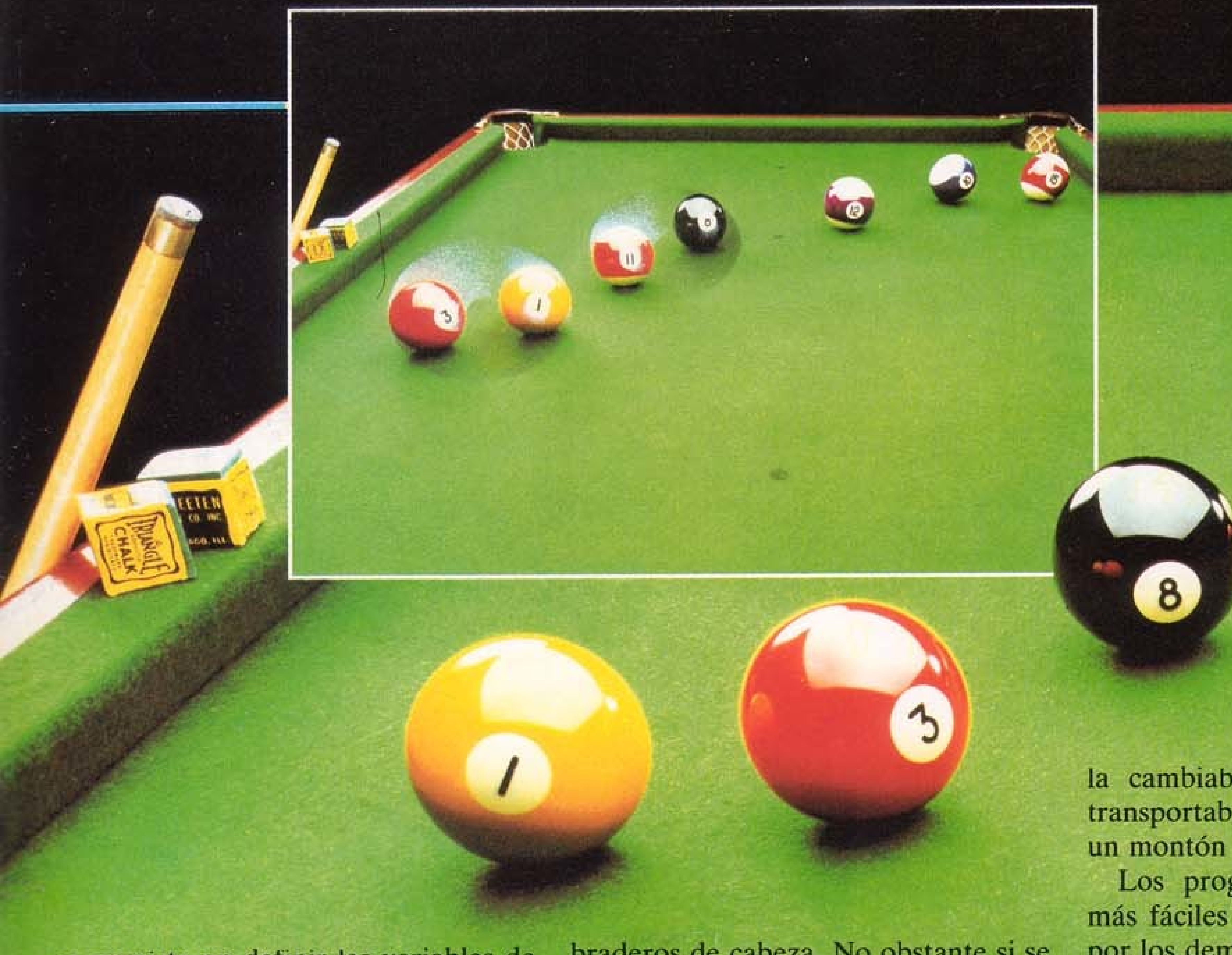
El programa anterior es una subrutina a la que hay que llamar desde el programa principal. Por ejemplo, para ordenar los elementos números 5 a 20, se podría llamar así a la subrutina:

```
100 LET N1=5:LET N2=20:GOSUB
     1000:REM METODO BURBUJA
```

Siempre tiene que haber además una corta sección de programa para introducir los elementos que quieras ordenar, y otra sección que presente la lista ya ordenada. Llegados a este punto, tienes que decidir la forma en que quieres que aparezca el resultado en la pantalla, haciendo incluso algún dibujo sobre papel si ello te resulta de ayuda.

PRUEBA DE LOS MODULOS

Cada uno de los módulos de tu diseño original terminará por convertirse en una subrutina de tu programa. Este método de fraccionamiento del programa ayuda mucho durante la fase de pruebas, ya que cada uno de los módulos puede ser probado y depurado separadamente. La idea bási-



ca consiste en definir las variables de entrada, llamar a la subrutina y examinar los resultados. Volviendo a la rutina de ordenación por el método de la burbuja, puedes hacer una prueba como la siguiente:

```
8 INPUT "NUMERO DE ELEMENTOS
  ";N
10 DIM A$(N,10)
12 PRINT "INTRODUCE LOS
   ELEMENTOS EN LA MATRIZ"
14 FOR I=1 TO N:INPUT A$(I):
   NEXT I
16 INPUT "RANGO A ORDENAR ";
   N1,N2
18 GOSUB 1000
20 PRINT "LISTA ORDENADA:"
22 FOR I=1 TO N:PRINT A$(I):
   NEXT I
24 GOTO 16
```

PONIENDOLO TODO JUNTO

Finalmente llega el momento de enlazar todos los módulos y probar el programa como un todo completo. A esto se le llama integración del programa. Si le has dedicado el debido tiempo y cuidado a las anteriores fases del programa, este proceso de integración no debe producirte demasiados quebraderos de cabeza. No obstante si se producen problemas, tienes que volver a comprobar todos los módulos sospechosos y modificarlos si es preciso. Al final has de tener un programa perfectamente estructurado que haga exactamente lo que tú quieras.

Para refresco de la memoria, aquí tienes las reglas que te conducirán a escribir un programa estructurado:

1.- Escribe una descripción general del programa.

2.- Fracciónala en módulos a todos los niveles que sea necesario.

3.- Para cada módulo, dibuja un diagrama de flujo y define las variables de entrada y salida, así como cualquier otros efectos, tales como la forma de presentación en pantalla.

4.- Escribe los programas correspondientes a cada uno de los módulos, utilizando las estructuras descritas en la parte 1.

Prueba cada uno de los módulos, dándole unas entradas y comprobando los resultados.

6.- Enlaza todos los módulos y realiza la prueba del programa completo: ¡Tiene que funcionar!

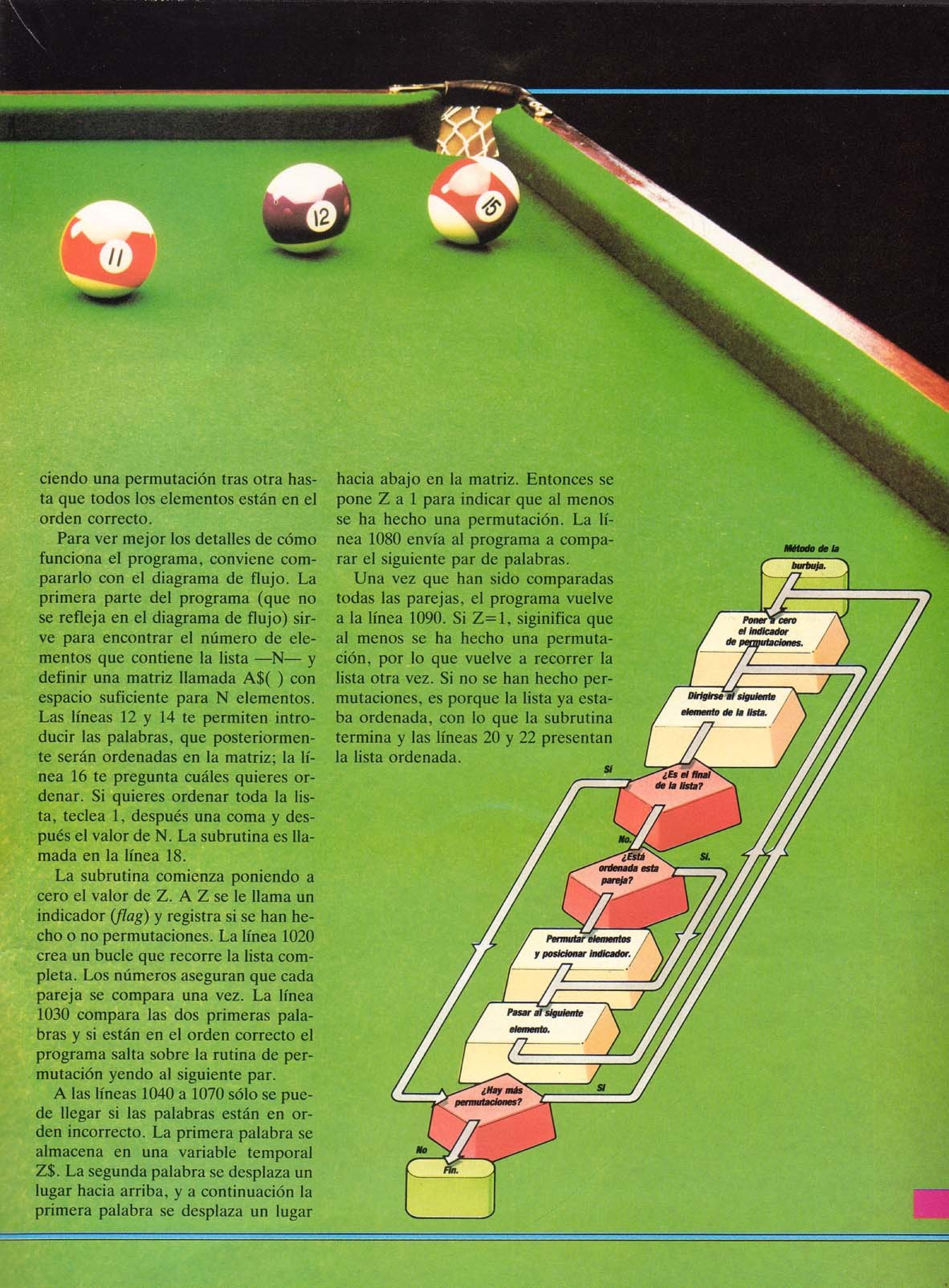
En caso de que te hayas olvidado de las razones que aconsejan que te tomes todo este trabajo, recuerda que son la legibilidad, la comprobabilidad, la cambiabilidad, la fiabilidad y la transportabilidad; como puedes ver, un montón de «habilidades».

Los programas estructurados son más fáciles de seguir por tí mismo y por los demás. Son más fáciles de depurar y de modificar. Es más probable que funcionen bien y resultan más fáciles de adaptar para que corran en otros ordenadores. Si tienes interés en cualquiera de estos resultados, tienes que pasarte a las técnicas estructuradas.

FUNCIONAMIENTO DEL METODO DE LA BURBUJA

Hemos utilizado el método de ordenación de la burbuja para ilustrar la forma en que se puede construir y probar un módulo antes de enlazarlo con el programa principal. Las rutinas de ordenación son muy útiles en cualquier tipo de programas. Esta rutina sirve para clasificar palabras por orden alfabético, pero podría utilizarse igualmente para ordenar números por orden creciente o decreciente. No tendrías más que cambiar las cadenas variables Z$ por Z y la matriz A$( ) por A( ).

El ordenador lee la lista completa de elementos y los va comparando dos a dos. Si están en el orden correcto los deja como estaban, mientras que si están desordenados los permuta. Continúa haciendo esto por toda la lista ha-



ciendo una permutación tras otra hasta que todos los elementos están en el orden correcto.

Para ver mejor los detalles de cómo funciona el programa, conviene compararlo con el diagrama de flujo. La primera parte del programa (que no se refleja en el diagrama de flujo) sirve para encontrar el número de elementos que contiene la lista —N— y definir una matriz llamada A$( ) con espacio suficiente para N elementos. Las líneas 12 y 14 te permiten introducir las palabras, que posteriormente serán ordenadas en la matriz; la línea 16 te pregunta cuáles quieres ordenar. Si quieres ordenar toda la lista, teclea 1, después una coma y después el valor de N. La subrutina es llamada en la línea 18.

La subrutina comienza poniendo a cero el valor de Z. A Z se le llama un indicador (*flag*) y registra si se han hecho o no permutaciones. La línea 1020 crea un bucle que recorre la lista completa. Los números aseguran que cada pareja se compara una vez. La línea 1030 compara las dos primeras palabras y si están en el orden correcto el programa salta sobre la rutina de permutación yendo al siguiente par.

A las líneas 1040 a 1070 sólo se puede llegar si las palabras están en orden incorrecto. La primera palabra se almacena en una variable temporal Z$. La segunda palabra se desplaza un lugar hacia arriba, y a continuación la primera palabra se desplaza un lugar hacia abajo en la matriz. Entonces se pone Z a 1 para indicar que al menos se ha hecho una permutación. La línea 1080 envía al programa a comparar el siguiente par de palabras.

Una vez que han sido comparadas todas las parejas, el programa vuelve a la línea 1090. Si Z=1, siginifica que al menos se ha hecho una permutación, por lo que vuelve a recorrer la lista otra vez. Si no se han hecho permutaciones, es porque la lista ya estaba ordenada, con lo que la subrutina termina y las líneas 20 y 22 presentan la lista ordenada.

¡PARTICIPA EN INPUT!



BASES

PROGRAMAS: Una vez desarrollado tu programa, que debe ser original y no haber sido enviado a ninguna otra publicación, puedes enviárnoslo aquí grabado en cassette, diskette o microdrive. Es preferible que vaya acompañado por un listado de impresora, pero no es imprescindible.

El programa habrá de venir acompañado por un texto que aclare cuál es su objetivo, el modo de funcionamiento y una explicación del cometido que cumplen las distintas rutinas que lo componen. El texto se presentará en papel de tamaño folio y mecanografiado a dos espacios. No importa que la redacción no sea muy clara y cuidada; nuestro equipo de expertos se encargará de proporcionarle la forma más atractiva posible.

ARTICULOS E IDEAS: Se aplica lo anteriormente dicho para los textos que acompañan a los programas; es decir, conviene detallar al máximo lo que desees que aparezca publicado en la revista, de la manera que te gustaría que otra persona hubiera explicado eso mismo.

UN JURADO propio decidirá en cada momento qué colaboraciones reúnen los requisitos adecuados para su publicación, y evaluará la cuantía del premio en metálico al que se hagan acreedoras.

No olvidéis indicar claramente para qué ordenador está preparado el material, así como vuestro nombre y dirección y, cuando sea posible, un teléfono de contacto. Entre todos los trabajos recibidos durante cada mes SORTEAREMOS:

- **Un premio de 50.000 ptas.**
- **Un premio de 25.000 ptas.**
- **Un premio de 10.000 ptas.**

en material microinformático a elegir por los afortunados.

¡No os desaniméis!, por muy simples o complejas que puedan parecer vuestras ideas, todas serán revisadas con el máximo interés.

INPUT SINCLAIR
Paseo de la Castellana, 93. Planta 14
28046 MADRID

NOTA: INPUT no se responsabiliza de la devolución del material que no vaya acompañado por un sobre adecuado con el franqueo correspondiente.



LOS MEJORES DE INPUT SINCLAIR

PUESTO	TITULO	PORCENTAJE
1.°	Commando	23,3 %
2.°	Green Beret	17,4 %
3.°	Rambo	12,1 %
4.°	Winter Games	9,5 %
5.°	Saboteur	8,2 %
6.°	Movie	7,9 %
7.°	Skyfox	7,7 %
8.°	Phantomas	5,3 %
9.°	Sir Fred	4,8 %
10.°	Batman	3,8 %
		100 %

Para la confección de esta relación únicamente se han tenido en cuenta las votaciones enviadas por nuestros lectores de acuerdo con la sección «Los Mejores de Input».

Noviembre de 1986.



EL ULTIMO HOMBRE EN VIETNAM







DATOS GENERALES

TITULO The Last Man of Vietnam

FABRICANTE Alligata

ORDENADOR Spectrum 48

TEMA DEL PROGRAMA
Bélico

CALIFICACION (Sobre 10 ptos.)

ORIGINALIDAD	3
INTERES	7
GRAFICOS	7
COLOR	6
SONIDO	8
TOTAL	31

The Last Man of Vietnam o **Who Dares Wins II** (quien arriesga gana), es casi, casi, una réplica de **Comando**, sólo que más violento si cabe.

Además de pasar varias fases sucesivas en territorio enemigo, tendrás que ir liberando a tus compañeros capturados. Los encontrarás atados a un árbol, a punto de ser fusilados por un norvietnamita sin escrúpulos, y agradecerán efusivamente tu gesto si consigues salvarles la vida.

Naturalmente, si tienes prisa puedes pasar de largo y dejarles morir, pero eso supone perder unos cuantos valiosos puntos.

Los gráficos son regulares, y la originalidad escasa, pero a pesar de todo no se puede negar que el programa es divertido.

Terminaremos insistiendo en su excesivo parecido con respecto al **Comando** (no queremos entrar en detalles), y recomendando que se tenga presente este dato a la hora de adquirir el juego.

****

SHOW JUMPING

DATOS GENERALES

TITULO Show Jumping

FABRICANTE Alligata

ORDENADOR Spectrum 48

TEMA DEL PROGRAMA
Hípica

CALIFICACION (Sobre 10 ptos.)

ORIGINALIDAD	9
INTERES	7
GRAFICOS	6
COLOR	6
SONIDO	7
TOTAL	35





Sabemos que un programa de este tema, por elevada que sea su calidad gráfica, lo tiene muy difícil para despertar el interés de quienes se aburren con la **Hípica**.

No obstante, desde aquí advertimos que **Show Jumping** es un programa que no puede aburrir ni al más exigente de los usuarios.

Admite hasta un máximo de **ocho jugadores**, se puede seleccionar el nivel de dificultad y también es posible redefinir las teclas. Los gráficos, aunque no son especialmente vistosos, tienen un nivel más que aceptable, y el grado de interés no deja nada que desear.

Al principio cuesta mucho saltar correctamente una sola valla, pero a medida que se va tomando práctica esta dificultad disminuye rápidamente. El secreto está en enfilar correctamente y calcular el ángulo adecuado.

En definitiva un buen programa.

PASEO ESPACIAL

DATOS GENERALES

TITULO Space Walk

FABRICANTE Mastertronic

ORDENADOR Spectrum 16K/48K

TEMA DEL PROGRAMA
Rescate Espacial

CALIFICACION (Sobre 10 ptos.)

ORIGINALIDAD	5
INTERES	6
GRAFICOS	5
COLOR	5
SONIDO	4
TOTAL	25

Suponemos que todavía quedará por ahí algún 16K sin ampliación de memoria, y podemos imaginarnos a su desesperado dueño buscando programas aptos para la corta RAM de su micro.
Pues bien, por fin alguien se ha acordado de aquel segundo modelo de **Sinclair** con el que muchos empezamos a programar, y que ahora, con el **128K** y el **PLUS 2** en candelero, casi nadie recuerda.
Aunque no le hemos dado altas calificaciones a **Space Walk**, si tenemos en cuenta que su listado es inferior a la mitad de 48K, la verdad es que poco más se puede pedir.
El argumento es bastante simple: un astronauta cuya misión consiste en rescatar satélites a la deriva, se enfrenta a una multitud de seres cósmicos.
El programa, naturalmente, también es apto para 48K, pero lo recomendamos principalmente para los micros de 16K.





****

CON-QUEST

DATOS GENERALES

TITULO Con-Quest

FABRICANTE Mastertronic

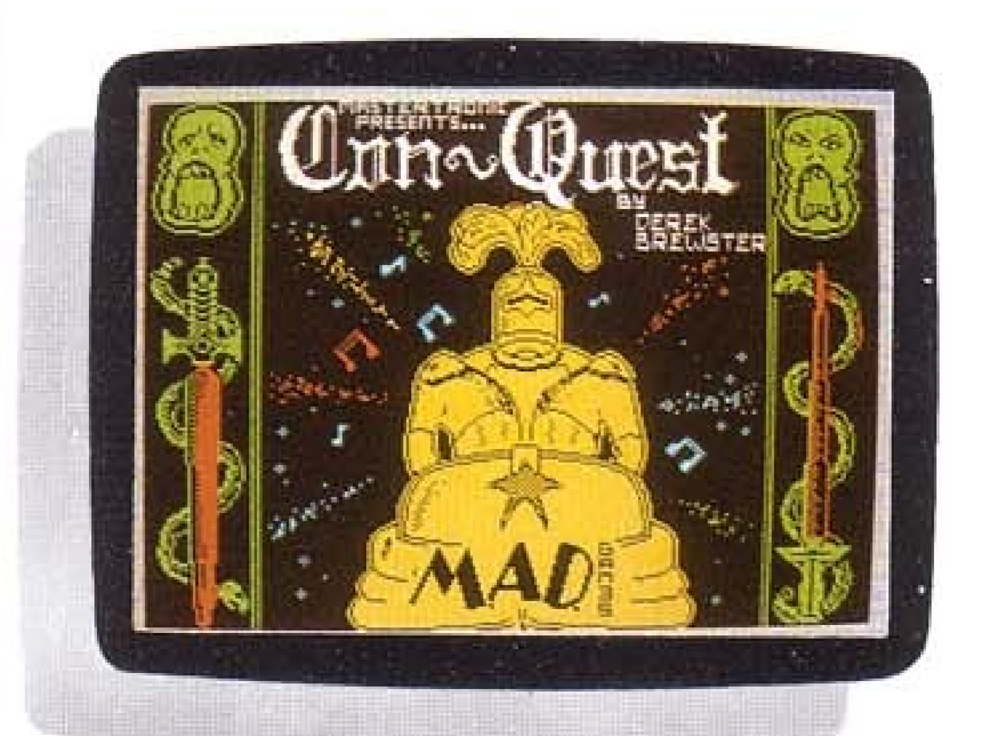
ORDENADOR Spectrum 48K

TEMA DEL PROGRAMA
Castillo Encantado

CALIFICACION (Sobre 10 ptos.)

ORIGINALIDAD	7
INTERES	6
GRAFICOS	6
COLOR	6
SONIDO	5
TOTAL	30

Con-Quest es una aventura en tres dimensiones que transcurre en el interior de un castillo supuestamente deshabitado. Decimos «supuestamente», porque lo primero que descubre **Oscar**, su nuevo propietario, es la molesta presencia de centenares de bichos y criaturas infernales, capaces de hacer huir al más templado. En otras palabras, quien le vendió el castillo le timó. Pero ya es demasiado tarde para reclamaciones. Ahora la principal preocupación de **Oscar** es escapar de las llamas del infierno y salvar su alma. Nosotros nos hemos esforzado todo lo que hemos podido para ayudarle, pero no hemos conseguido llegar muy lejos, ya que el nivel de dificultad es alto.
El programa tiene un tratamiento gráfico correcto, pero le falla el interés. Su argumento es insulso y un poco trasnochado. No obstante, en líneas generales está bien y seguro que entusiasmará a los amantes de esta clase de juegos.
Para que os hagáis una idea más aproximada, os diremos que se parece a aquel programa que se llamaba **Atic-Atac**.





STRIKE FORCE HARRIER

La verdad es que este programa nos ha dejado tan impresionados, que no sabemos por dónde empezar a comentarlo. Por el momento, lo único que tenemos claro es la calificación final: sensacional. Mandamos nuestras más efusivas felicitaciones a los señores de **Mirrosoft** (y hasta les perdonamos lo del **Dynamite Dan II**), y a los afortunados que ya sean propietarios







DATOS GENERALES

TITULO Strike Force Harrier

FABRICANTE Mirrorsoft

ORDENADOR Spectrum 48

TEMA DEL PROGRAMA Simulador

CALIFICACION (Sobre 10 ptos.)

ORIGINALIDAD	8
INTERES	9
GRAFICOS	9
COLOR	8
SONIDO	7
TOTAL	41

de un ejemplar de **Strike Force Harrier**. Quizás puede parecer que nos estamos excediendo, así que vamos a dejarnos de alabanzas y pasemos a los hechos, para que los escépticos se convenzan por sí mismos.
Se acabaron los paisajes sin relieve, los fondos planos y los gráficos austeros. En **Strike Force Harrier** hay incluso nubecitas, amén de escarpadas montañas, varias especies de arbolitos, diversos tipos de armas terrestres, y naturalmente, mortíferos cazas enemigos. Felizmente, tampoco se ha descuidado el color (error bastanteshabitual en estos casos), ni los detalles del interior de la carlinga. Esto, en cuanto a la parte gráfica (y no lo hemos dicho todo).
Por lo demás, existen tantas funciones como teclas tiene el ordenador, ni una menos, pero todas ellas son de sencillo manejo. Se reproducen con toda fidelidad no sólo las peculiares características técnicas del **Harrier** sino también sus **tácticas de combate** (ilustradas en un folleto de instrucciones que nada tiene que envidiar a un manual de aeronaútica), y su famosa modalidad de **despegue «vertical»**.
En fin, no podemos asegurarte que vayas a vivir las mismas experiencias que los pilotos de la **RAF** en las Malvinas, pero casi, casi.

****

MERMAID MADNESS

La verdad es que ya estamos un poco hartos de héroes que rescatan princesas, y de naves que destruyen invasores del espacio. Lo que necesitamos es algo diferente, con imaginación, algo como por ejemplo lo siguiente:
Obesa sirenita de ciento veinte años, persigue a apuesto submarinista para seducirle con ciento veinte kilos de «encantos». Apuesto submarinista huye despavorido hasta el muelle más próximo, salta al agua y se pierde en las profundidades del óceano. Obesa sirenita le busca infatigable. Apuesto submarinista queda atrapado en un barco hundido, y su bombona de oxígeno se agota por momentos. Obesa sirenita se apresta a rescatarle.
Tal es el argumento de **Mermaid Madness**. Originalidad: un nueve. Desgraciadamente, no podemos dar una nota tan alta como ésta a los gráficos, debido a un pequeño fallo que se puede apreciar cuando la sirenita queda atrapada entre dos pantallas. El color, no obstante, no está nada mal, y el grado de interés es bastante alto.
Si diéramos un premio al argumento más original, **Mermaid Madness** estaría entre los principales candidatos. Aunque sus gráficos son modestos, están realizados con mucha imaginación (especialmente

DATOS GENERALES

TITULO Mermaid Madness

FABRICANTE Electric Dreams

ORDENADOR Spectrum 48K

TEMA DEL PROGRAMA
Comedia Submarina

CALIFICACION (Sobre 10 ptos.)

ORIGINALIDAD	**9**
INTERES	**8**
GRAFICOS	**7**
COLOR	**7**
SONIDO	**4**
TOTAL	**35**

las «grasas» de la sirenita), y cumplen sobradamente con su cometido. Por otra parte, el bajo nivel de dificultad contribuye a hacer aún más atractivo el programa desde el primer momento.
En suma, **Mermaid Madness** merece una valoración general positiva, aunque debemos advertir a los señores de **Electric Dreams** que a estas alturas no se pueden tolerar alegremente determinados errores.





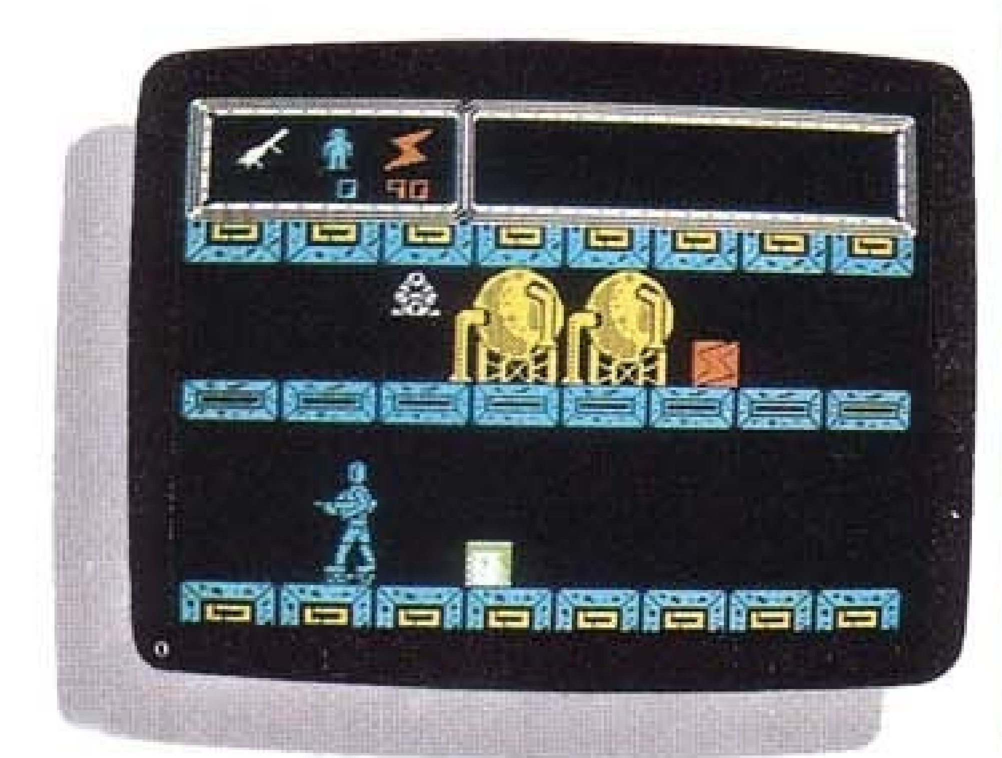


★★★★★★★★★★★★★★★★★★★★★★★★★★★

TUJAD

Tujad es un programa que destaca, ante todo, por su originalidad. Aunque el tema no es especialmente novedoso (más bien al contrario), su tratamiento si que está dotado de una cierta peculiaridad, que podríamos definir así:
Tanto el desarrollo gráfico, como el planteamiento general del juego, son totalmente difernetes a lo que hemos visto hasta ahora. Nadie mejor que vosotros sabe que pocos programas pueden presumir de ello, pero desgraciadamente, sólo recordamos tres o cuatro juegos que, aparte de esta máxima originalidad, tuvieran también todo lo demás, y **Tujad** no está entre ellos.
Es una verdadera pena porque se trata de un programa excelente, del que no se ha sabido sacar todo el partido.
El tema (por cierto, muy parecido al de **Equinox**), puede resumirse en una central nuclear con problemas, y un androide que tiene que arreglárselas para solucionarlos. En concreto, se trata de hallar y reunir las diversas partes que forman un circuito impreso, y reparar de esta manera una peligrosa avería en el supercomputador central. Nuestro androide dispone de diversos tipos de armas, cada una de ellas adecuada a una clase diferente de enemigos. En el interior del reactor, pueden encontrarse suministros adicionales (la munición se agota rápidamente), y con un poco de suerte, alguna que otra vida extra.
En los momentos de máximo peligro, se puede salir del paso usando un escudo de defensa, pero la verdad es que **Tujad** no tiene muchos momentos de ese tipo. Se trata de un juego tranquilo, sin sobresaltos, cómodo de jugar.





DATOS GENERALES

TITULO Tujad

FABRICANTE Ariolasoft

ORDENADOR Spectrum 48K

TEMA DEL PROGRAMA
Avería en reactor nuclear

CALIFICACION (Sobre 10 ptos.)

ORIGINALIDAD	8
INTERES	8
GRAFICOS	8
COLOR	7
SONIDO	5
TOTAL	36

★★★★★★★★★★★★★★★★★★★★★★★★★★★★★

ROCKMAN

En más de una ocasión, hemos comprobado hasta qué punto pueden llegar a burlarse los programadores británicos de las «supuestas» limitaciones del **Spectrum**.
Comentarios como «parece mentira que esto quepa en 48K», han quedado ya fuera de lugar. Nada nos sorprende.
Sin embargo, la resolución gráfica, la velocidad de ejecución, las interrupciones, los *sprites*, y todos los elementos que han sido desarrollados en estos últimos años, han tenido y siguen teniendo un «pariente pobre» tristemente ignorado por todos: **El Sonido**.
Y ha tenido que ser precisamente un español quien viniera a demostrarnos que, al igual que se han sabido superar las limitaciones gráficas, también es posible saltarse a la torera la cacareada «pobreza técnica» del sonido.
El programa en cuestón es **Rockman**, y el autor de la proeza **Javier Martín**.
El juego, además de poseer unos gráficos magníficos y un argumento con elevado grado de interés, se presenta a sí mismo con una **voz humana digitalizada**, perfectamente audible, y casi tan clara como si la escucháramos a través de una radio.
Rockman es un programa sorprendente; que os recomendamos no sólo por sus increíbles efectos sonoros, sino también por la calidad de sus gráficos y argumento.

DATOS GENERALES

TITULO Rockman

FABRICANTE Alligata

ORDENADOR Spectrum 48

TEMA DEL PROGRAMA
Buscando tesoros

CALIFICACION (Sobre 10 ptos.)

ORIGINALIDAD	7
INTERES	9
GRAFICOS	8
COLOR	8
SONIDO	10
TOTAL	42







****

URIDIUM

A finales de este verano, la firma **Hewson** presentó en Inglaterra sus dos últimos programas estrella, causando un fuerte impacto en la severa crítica británica. Uno de ellos (algunos dicen que el mejor de los dos), es **Uridium**, un sensacional arcade con una calidad gráfica sólo comparable a los videojuegos de las máquinas tragaperras.
En la carátula del programa, encontramos un reclamo que dice, textualmente: «**Uridium** es el juego de arcade más rápido y adictivo que jamás hayas visto en tu ordenador». Matizando un poco, añadiríamos que no sólo posee una rapidez excepcional, sino que además ésta no se ha conseguido a expensas de saltos como suele ser habitual. Por el contrario, nos maravillamos contemplando un *scroll* impecable y unos desplazamientos extraordinariamente suaves.
Por otra parte , el color y el diseño parecen burlarse de la limitada capacidad de resolución gráfica del **Spectrum**, creando un efecto de sorprendente realismo.
Pocas veces hemos visto un arcade de la calidad de **Uridium**.

DATOS GENERALES

TITULO Uridium

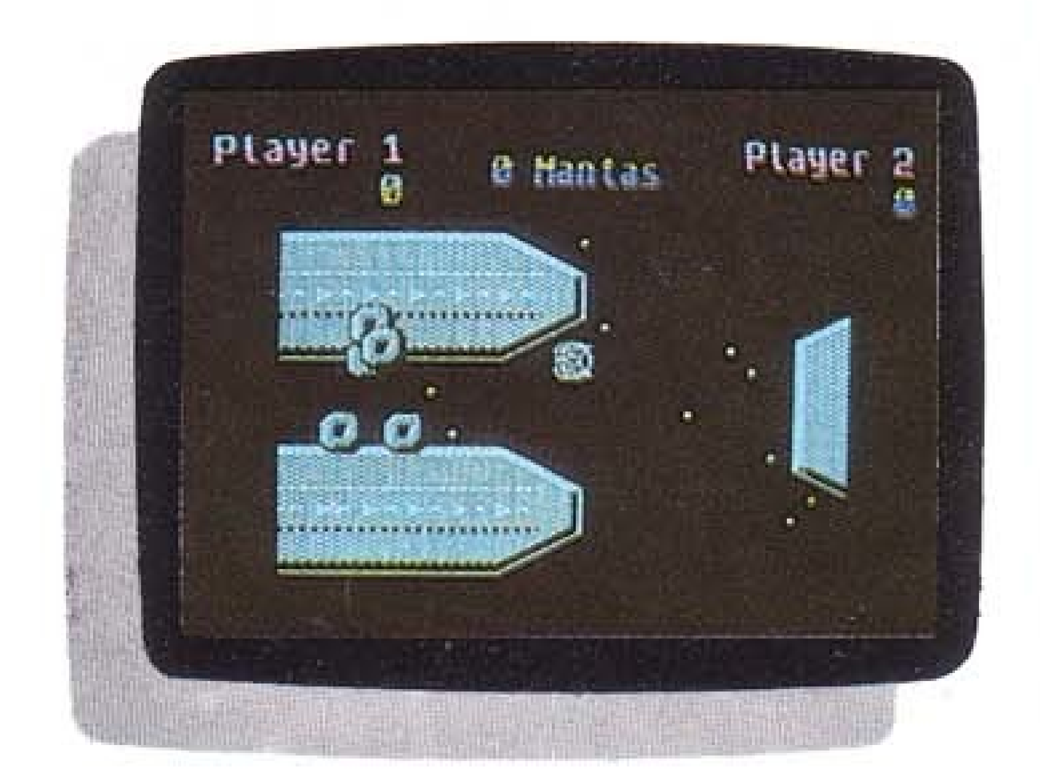
FABRICANTE Hewson

ORDENADOR Spectrum 48K

TEMA DEL PROGRAMA
Arcade

CALIFICACION (Sobre 10 ptos.)

ORIGINALIDAD	8
INTERES	8
GRAFICOS	9
COLOR	9
SONIDO	6
TOTAL	40



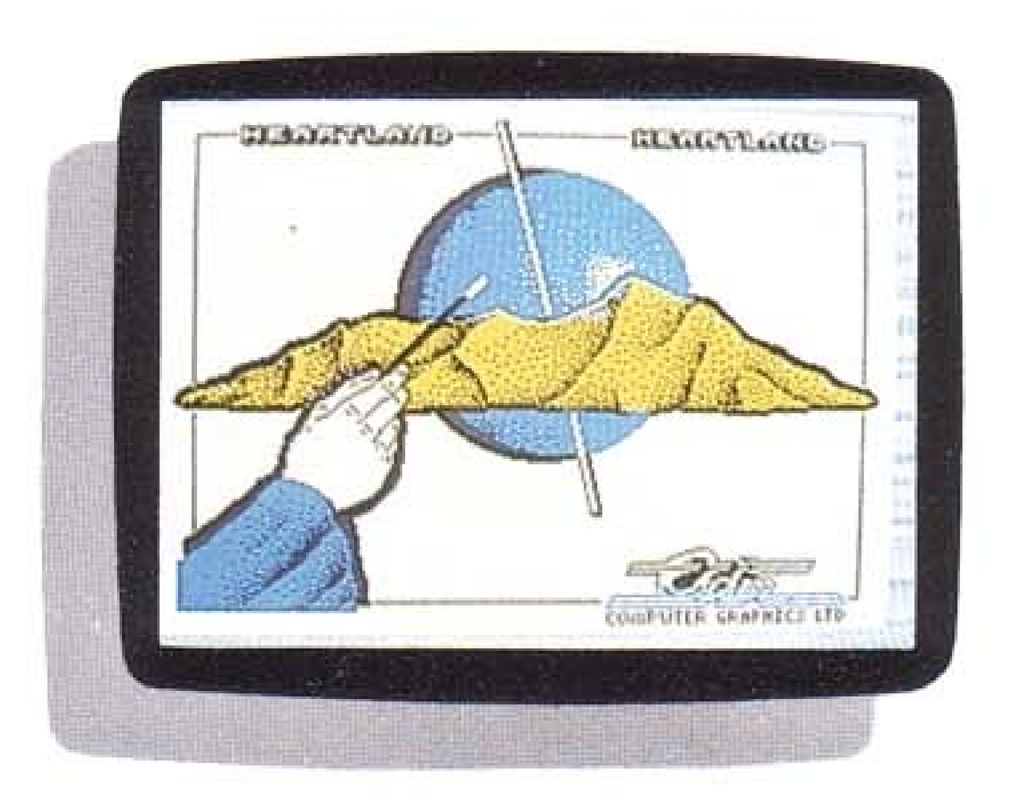


****

HEARTLAND

Entre las mejores novedades presentadas en el **Personal Computer Show** de Londres, celebrado el pasado mes de Septiembre, brilló con luz propia una de las últimas creaciones de **Odin**: **Heartland**.
Se trata de un magnífico programa desarrollado recientemente por los creadores de **Nodes of Yesod**, que está obteniendo un éxito clamoroso en el Reino Unido. La verdad es que no le faltan motivos a los usuarios británicos para adquirirlo: un repertorio gráfico original y abundante, con *sprites* sorprendentemente impecables, y un elevadísimo grado de interés, son razones más que suficientes ¿No?
Es difícil resumir el contenido de un programa como **Heartland** en unas pocas líneas, y por otra parte, este es uno de esos juegos que hay que ver para creer, pero vamos a tratar de intentarlo. Después, terminaremos el comentario con un pequeño apartado que os puede ser de gran utilidad.
Un buen día, descubres en el ático de tu casa un viejo libro que cuenta la historia de **Heartland**, un mundo fantástico en el que se libra una

terrible batalla entre las fuerzas del bien, y las hordas demoníacas de un diantre llamado **Midas**. Enfrascado en su lectura, caes en un profundo sueño que te transporta al interior del libro, y despiertas rodeado de extraños personajes y enigmáticos lugares, con una idea obsesiva en tu cerebro: encontrar las páginas finales del libro, en las que se describe el resultado final de la guerra.



Para dificultar tu misión, el diablo **Midas** ha colocado en el camino que has de seguir varias páginas falsas que cuentan la derrota de **Heartland**, pero su color negro te permitirá distinguirlas fácilmente de las auténticas. Naturalmente, no será éste el único obstáculo al que deberás enfrentarte. Cientos de seres infernales te arrebatarán parte de tu energía, y profundos abismos



jalonarán tu difícil camino.
Para culminar con éxito la aventura, dispones de tres tipos de armas con diferentes grados de efectividad (sombreros, espadas, y bolas de fuego). Podrás recogerlas cuando aparezcan flotando por el borde superior de la pantalla.
El nivel de energía es un elemento importantísimo en el juego. A medida que tus reservas se vayan agotando, aparecerá una imagen del rostro del diablo **Midas**. Cuando dicha imagen se complete, y comience a parpadear, tus segundos estarán contados. Para mantener tu nivel de energía y reponer las fuerzas perdidas deberás alimentarte de las burbujas que vayas encontrando, y evitar el roce de tus enemigos.
El programa consta de seis niveles, cada uno de ellos con un repertorio gráfico diferente y con distintos tipos de escenarios. La comunicación entre las diversas zonas de cada nivel no tiene lugar a través de *scroll*, sino que se produce un salto de una pantalla a otra, propiciando una mayor rapidez y guardando el misterio de lo que se va a encontrar más allá hasta el último momento.



DATOS GENERALES

TITULO Heartland

FABRICANTE Odin

ORDENADOR Spectrum 48K

TEMA DEL PROGRAMA Aventura Fantástica

CALIFICACION (Sobre 10 ptos.)

ORIGINALIDAD	9
INTERES	9
GRAFICOS	9
COLOR	7
SONIDO	8
TOTAL	42

ALGUNOS CONSEJOS

Nunca salgas de una pantalla caminando, pues es posible que el suelo se acabe y caigas a un abismo. Si lo haces saltando, en ese caso rebotarías en el borde sin sufrir ningun daño.



El arma más eficaz es la bola de fuego. Con un solo impacto, podrás destruir a cualquier enemigo, mientras que con las espadas y los sombreros son necesarios dos y tres respectivamente.
Vigila siempre tu nivel de energía. Cuando éste sea muy bajo, espera unos segundos en algún lugar tranquilo, y pronto aparecerán las burbujas.



¿Cómo saber dónde se encuentran las páginas del libro? Cuando veas que éste parpadea en la parte superior de la pantalla, quédate donde estés, acaba con tus enemigos y espera. Al cabo de unos segundos verás una de las páginas. Recógela, y vuelve rápidamente al lugar donde estaba la cama. Colócate sobre ella y oprime el botón de disparo. Una vez hecho esto, pasarás automáticamente a la fase siguiente. Ten siempre muy en cuenta que dispones de un tiempo límite para cada nivel. Un reloj de arena te indicará en todo momento cuánto tiempo te resta.
Nada más comenzar la partida, muévete siempre hacia la izquierda y encontrarás el libro.
Cuando quieras pasar de una zona a otra, colócate sobre un arco, puerta

o gruta. Si los gráficos pasan por delante de la entrada, colocáte de espaldas y pulsa «P» (o mueve el *joystick* hacia adelante); si pasan por detrás, ponte de frente y pulsa «L» (o mueve el *joystick* hacia atrás).
Después, para volver al lugar de partida, debes actuar de igual manera.
Una última advertencia: Nunca dispares al libro!!

★★★★★★★★★★★★★★★★★★★★★★★★★★★

NIGHTMARE RALLY

Como todos esperábamos, la nueva orientación que dió la firma **Ocean** al contenido de sus programas, no ha significado un abandono definitivo de los temas deportivos. Prueba de ello es el lanzamiento de un excelente simulador que hizo su aparición en Inglaterra durante el verano, y que ahora por fin está disponible en España. Se trata de **Nightmare Rally**.

DATOS GENERALES

TITULO Nightmare Rally

FABRICANTE Ocean

ORDENADOR Spectrum 48K

TEMA DEL PROGRAMA Rallies

CALIFICACION (Sobre 10 ptos.)

ORIGINALIDAD	8
INTERES	8
GRAFICOS	7
COLOR	6
SONIDO	5
TOTAL	34



Sus autores han sabido dotar al programa de todos los elementos necesarios para reproducir fielmente la emoción y las durísimas condiciones de los rallies. En este sentido, el título no puede ser más elocuente (*Nightmare* quiere decir pesadilla).
En vuestro camino podréis encontrar espesas nieblas, ríos que deben cruzarse a la velocidad adecuada,



obstáculos que requieren dar marcha atrás, montículos-rampa para dar espectaculares saltos, irregularidades del terreno que afectan a la capacidad del frenado del vehículo y dañan a los neumáticos, depósitos de combustible, y algún que otro detalle más, que en este momento no recordamos. Por si con todo ello no hubiera suficiente, y para echarle un poco de imaginación al programa, se han añadido dos objetos mágicos, de propiedades desconocidas :**el Dolmen** y **el Iris parpadeante**.
La vista desde la cabina al exterior, ocupa las dos terceras partes de la pantalla. El tercio inferior se dedica a los gráficos de la consola, y en él se encuentran, además del volante, los siguientes indicadores: indicador de velocidad, cuenta-revoluciones, temperatura del motor, nivel del combustible, indicador de distancia, cronómetro y selector de marchas.
Lo único que falta para que este simulador sea «real como la vida misma», es salir del coche, abrir el capó, y reponer la correa del ventilador o mirar el nivel del aceite.
Finalmente, unas palabras sobre los gráficos. La sensación de velocidad y el efecto de aproximación de los objetos lejanos son impresionates, pero la vistosidad y el diseño de los gráficos como podéis apreciar en las pantallas que acompañan a este comentario, no van tan allá. No se puede pedir todo.

★★★★★★★★★★★★★★★★★★★★★★★★★★★★★


QUE NO TE FALTEN
LOS IMPRESCINDIBLES!!
ZAFI CHIP
SUZUKI
TT RACER
CASIO
SHELL
33
SUZUKI
Increíblemente Real!
PON A PRUEBA TU HABILIDAD Y DESTREZA EN LOS CIRCUITOS DE
Nurburgring
Silverstone Monza
Paul Ricard
Jarama
SPECTRUM AMSTRAD
AMSTRAD DISK
DYNAMITE DAN II
¡ EL TRIUNFO ¡
MIRRORSOFT
SPECTRUM AMSTRAD
AMSTRAD DISK
Editado, fabricado y distribuido en España bajo la garantía Zafiro. Todos los derechos reservados.
ZAFIRO SOFTWARE DIVISION
Paseo de la Castellana, 141 28046 Madrid
Tel. 459 30 04 Telex 22690 ZAFIR E



EL ZOCO

Estoy interesado en el intercambio de programas para el Spectrum 48K. También quisiera vender ZX-81 y un videojuego ATARI. Completos. Precios a convenir. Interesados escribir a:

José Alvarez Candedo
Manuel de Castro, 8, 2° izda.
Tel. (986) 20 99 42
Vigo (Pontevedra)

Vendo ordenador Sinclair QL y monitor de fósforo verde Philips BM 7552/00C, completamente nuevos, con cables, manuales en castellano, libros, programas, cartuchos microdrives... Precio a convenir. Extraordinaria oferta. Dirigirse a:

Luis Angel Gálvez Villar
Asociación Juvenil Aragonesa
Plaza de Sta. Cruz, 19
Tel. (976) 23 95 23
50003 Zaragoza

Ocasión. Vendo programas originales. Muy buenos. También vendo libros, tomo I de la enciclopedia Run, juegos de mesa y maquetas. Todo en muy buenas condiciones. Lo cambiaría por periféricos, preferiblemente impresora o microdrive/s con su interface I. Propongo un cambio equitativo. En caso de venta el precio es negociable. Interesados llamar o escribir a:

Angel Durán García
Pza. Fonsagrada, 5
Tel. 201 66 02 (preferiblemente de 11 a 2 mañanas)
28029 Madrid

Compro monitor (verde o color) para Spectrum. Precio a convenir. Interesados llamen a:

Fernando
Tel. 209 84 53
Barcelona

Intercambio y vendo juegos del Spectrum 48K. Tengo los mejores del mercado como el Green Beret, Rasputín, Quazatron, Ski fox, Turbosprit, etc.

Francisco Manrique Agredano
Avda. del Aeropuerto, 14-19-3
Tel. 41 48 65
14004 Córdoba

Cambio Spectrum 48K completo, con manuales en castellano, interface tipo Kempston, 9 cintas originales con instrucciones, 33 revistas la mayoría Microhobby. Todo por Commodore 64 con cassette. Negociables. Interesados llamar o escribir a:

José (hijo)
Gagos de Mendoza, 8, 7D
Tel. (986) 85 48 01
36001 Pontevedra

Se vende Spectrum Plus por 30.000 ptas. Se regala además los numeros del 1 al 8 de Input Sinclair. Y doy 10 programas comerciales a elegir en una lista de 100. Interesados llamar a:

Mari
Almazán S.A.
Tel. (975) 30 10 11

Estoy interesado en cambiar utilidades e información sobre el Spectrum 16 ó 48K.

Daniel
Tel. (965) 52 21 90
(horas de comida)

Vendo ZX-Spectrum 48K Plus. Muy buen estado (sólo 7 meses), manuales, cables de conexión y 20 juegos (Zorro, Comando, Les Flics, etc.) Todo por 30.000 ptas.

Juan José Escudero Cabrera
España, 25
San Agustín (Almería)

Vendo curso de informática completo de la importante firma E.C.C. valorado en 100.000 ptas., consta de 10 tomos de lecciones, en los que hay BASIC y COBOL; además incluye libros de cuestiones y respuestas para cada lección, todo ello va en una maleta, con material como reglas para diagramas, alfanuméricas, etc. Además al comprador regalo ordenador ZX-Spectrum 48K. Todo ello por la pequeña cantidad de 59.600 ptas. Llamar a:

Arantxa
Tel. (93) 801 08 03

Por cambio de sistema urge vender ordenador New Brain, modelo A, con 29K de ROM y 32K de RAM (ampliables hasta 2 megas), con fuente de alimentación, cables para cassette y monitor y manual en inglés; Vendo asimismo, monitor Fontec de fósforo verde de 12 pulgadas. Regalo cinta «DEMO» con guía en castellano, dos libros en inglés, cintas de utilidades (proceso de texto, contabilidad personal, base de datos, etc...) y de juegos (Ajedrez, El Muro, Batalla Espacial, etc.), por valor de más de 20.000 ptas. Precio indicativo: 45.000 ptas. Negociables. Interesados llamar y preguntar por:

Adolfo
Tel. (93) 319 00 47 (mañanas)
y (93) 32 20 49 (a partir de las 10 de la noche)

Vendo procesador de texto especial para GP-50-S o compatibles. Permite la impresión de textos en 64 columnas sin reducción de caracteres ni modificaciones en el hardware. Informa:

Manuel Cagiao
Apartado 2144
15080 La Coruña

Cambio programas para Spectrum. Poseo bastantes programas nuevos como Spindizzy, Samantha Fox, World Cup, Cauldron 2, Las tres luces de Claurung, Bounder, etc.

Javier González Alvarez
Puerto Pajares, 5, 6°D
Tel. (985) 39 83 28
(de 2 a 3 de la tarde y de 9 a 11)
Gijon (Asturias)

Vendo joystick, Quick Shot II, sin estrenar, por 1.000 ptas., o menos. Interesados escriban o llamen a:

David Escardó Torres
Rbla: La Pau, 34-36, piso 4°, 3ª
Tel. 815 04 62
Vilanova i la Geltrú
08800 Barcelona

Vendo Spectrum 48K. Poco uso. Con manuales, juegos, cinta de Horizontes, garantía y alimentador. Además regalo 3 cintas de Microhobby Cassette, 1 de Your Computer, 15 revistas de Microhobby semanal, 1 de Microhobby Especial, 2 de RUN, y 2 de INPUT. Todo por 22.500 ptas.

Rafael García
Obispo Velasco, 26, 3°
Tel. (947) 50 44 50
Aranda de Duero (Burgos)

Compro o cambio los siguientes juegos: Mickie, West Bank, Sport Winter I y II, Sir Fred, Gunfright, Atic-Atac, Robin of Sherwod,...etc. Interesados llamen o escriban a:

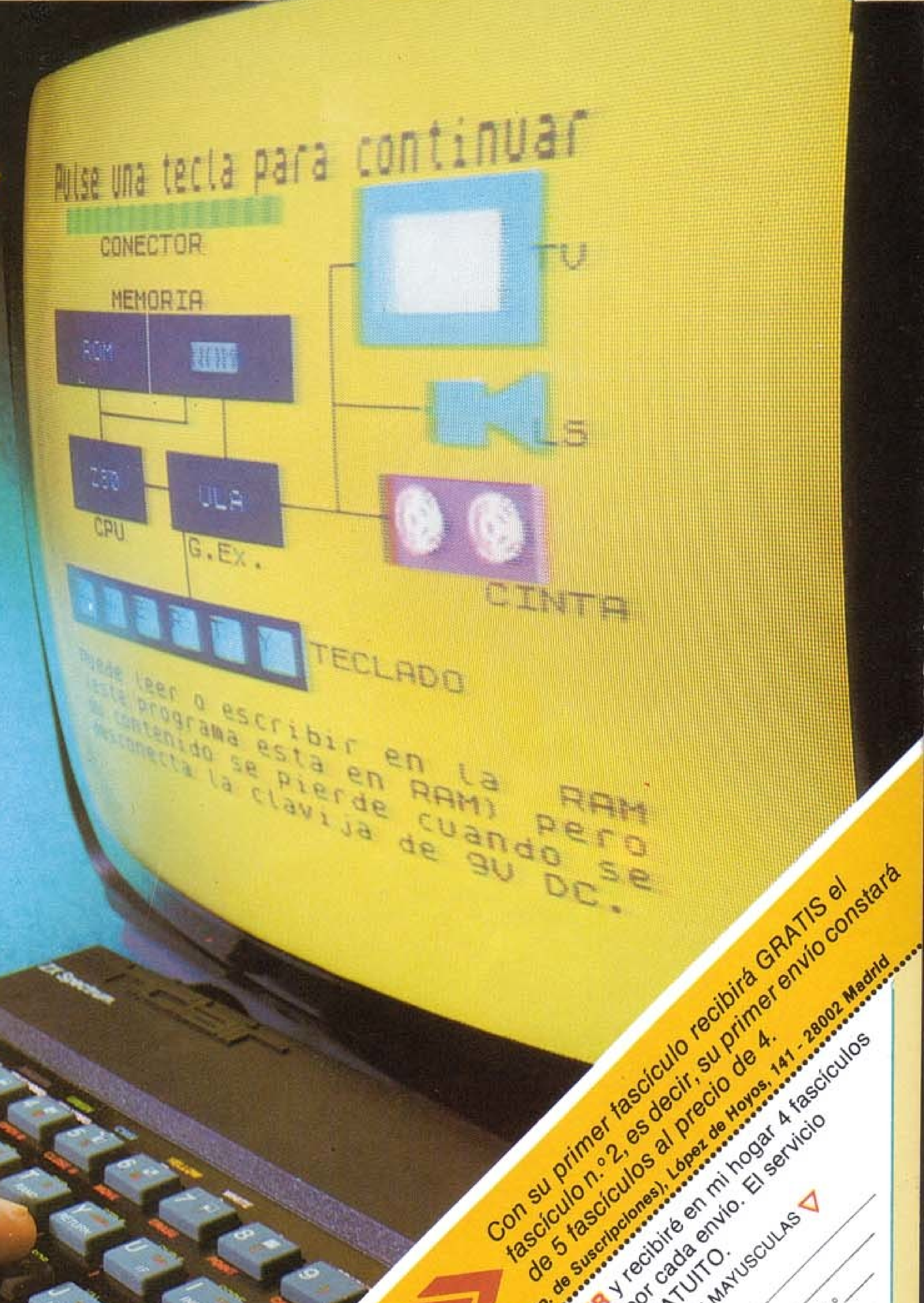
Andrés David Romero Marcos
Badajoz, Edif. Calipso 2°A
Tel. (952) 78 12 74 (de 7 a 9 de la tarde)
San Pedro de Alcántara
29670 Málaga

Intercambio, compro, vendo programas de E.A.O. para E.G.B. en Spectrum 48.

Javier Maestre Torregrosa
Av. Fernández Ladreda, 9 portal 6, 6°D
11006 Cádiz

MI COMPUTER

CURSO PRACTICO DEL ORDENADOR PERSONAL, EL MICRO Y EL MINIORDENADOR



Editorial Delta, S.A.

SORTEO PATROCINADO POR MICROUNO

****

La firma Microuno sortea 3 premios entre los lectores de INPUT Sinclair que enviéis una tarjeta postal conteniendo vuestros datos (nombre, apellidos, edad, dirección, ordenador...) y contando brevemente para que utilizáis el ordenador, si lo tenéis. La fecha tope de llegada es el próximo 15 de diciembre.

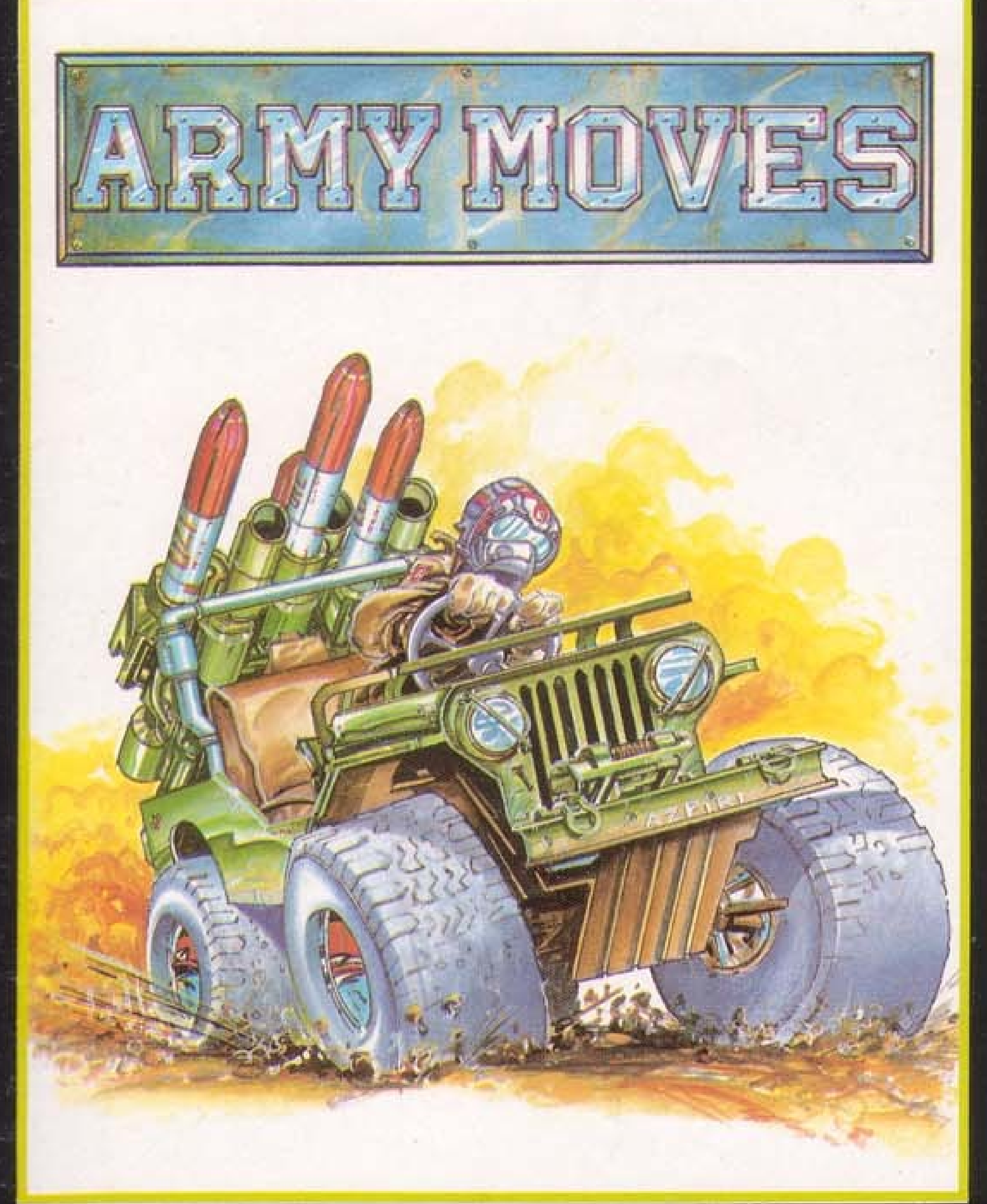
Los premios son:

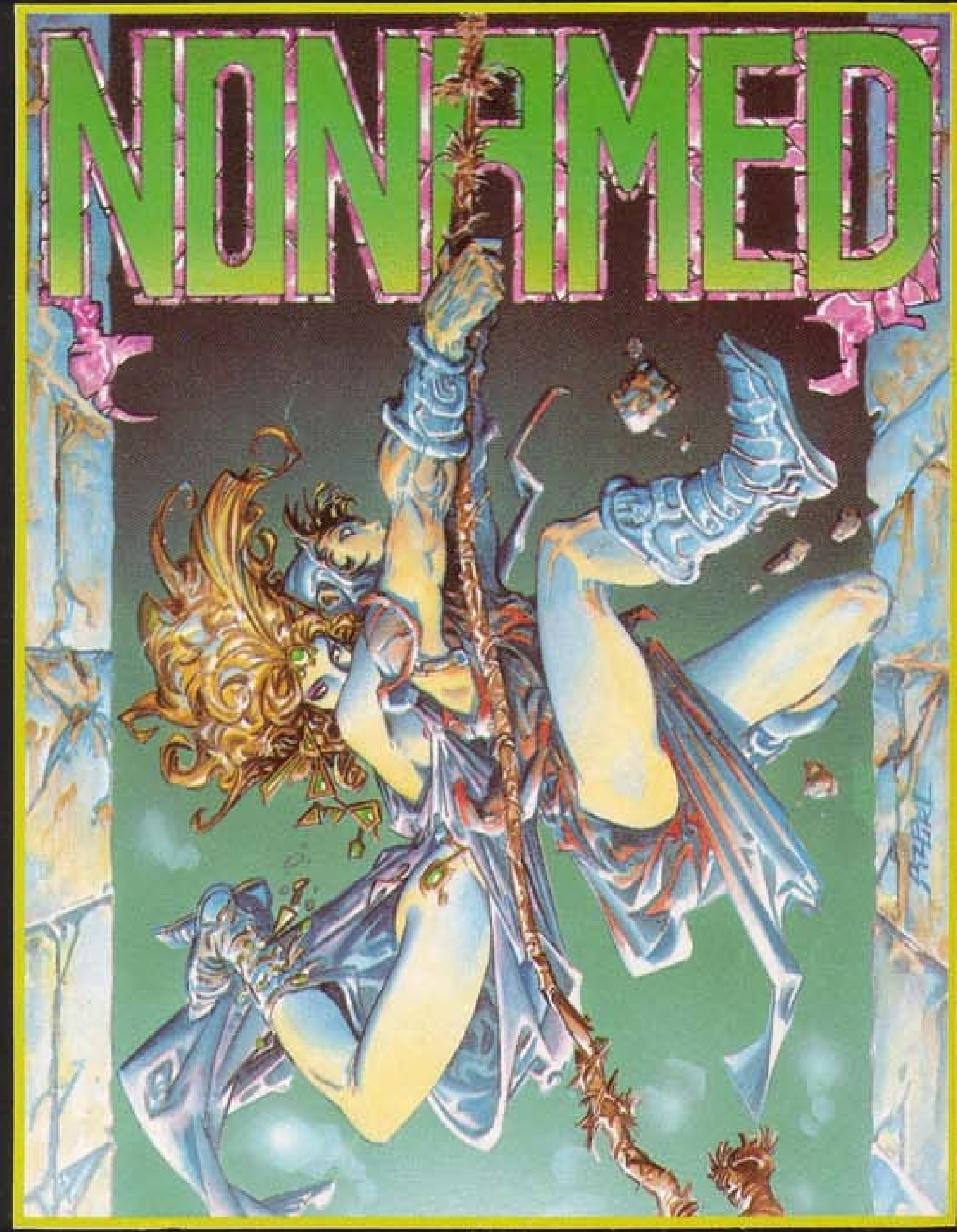
Un ordenador Amstrad 6128.
Un dispositivo Wafadrive.
Doce programas de juego a elegir.

Escribir a:

INPUT Sinclair. (Sorteo Microuno).
Po. Castellana, 93-Planta 14.
28046 MADRID

NUEVA ERA





AFTER THE WAR

ARMY MOVES — **SPECTRUM + 2 COMPATIBLES**

DERDHAL es un miembro del C.O.E., Cuerpo de Operaciones Especiales. Ha sido entrenado durante largos años para convertirse en un especialista y ahora es el primero de su promoción.
Puede atravesar las líneas enemigas por tierra, mar o aire, domina todas las técnicas de la guerra en la selva, conoce todas las armas y es un experto en explosivos.
ARMY MOVES, tres sistemas de combate distintos:
- *JEEP equipado con misiles tierra-aire.*
- *HELICOPTERO COBRA para la lucha en la jungla.*
- *SOLDADO COE miembro de un cuerpo de élite, entrenado en todas las técnicas conocidas para la guerra.*

NONAMED — **SPECTRUM + 2 COMPATIBLES**

Para ser caballero del rey no existe otro sistema. Tu obligación es superar la prueba, dominar el miedo, sufrir el rito. Tienes que encontrar la salida del castillo sin nombre donde te han encerrado.
En la búsqueda conocerás unos extraños ogros, que realmente no son más que otros intrépidos caballeros que no consiguieron encontrar la puerta del Castillo y fueron hechizados por el mago NILREM.

AFTER THE WAR: — **SPECTRUM + 2 COMPATIBLES**

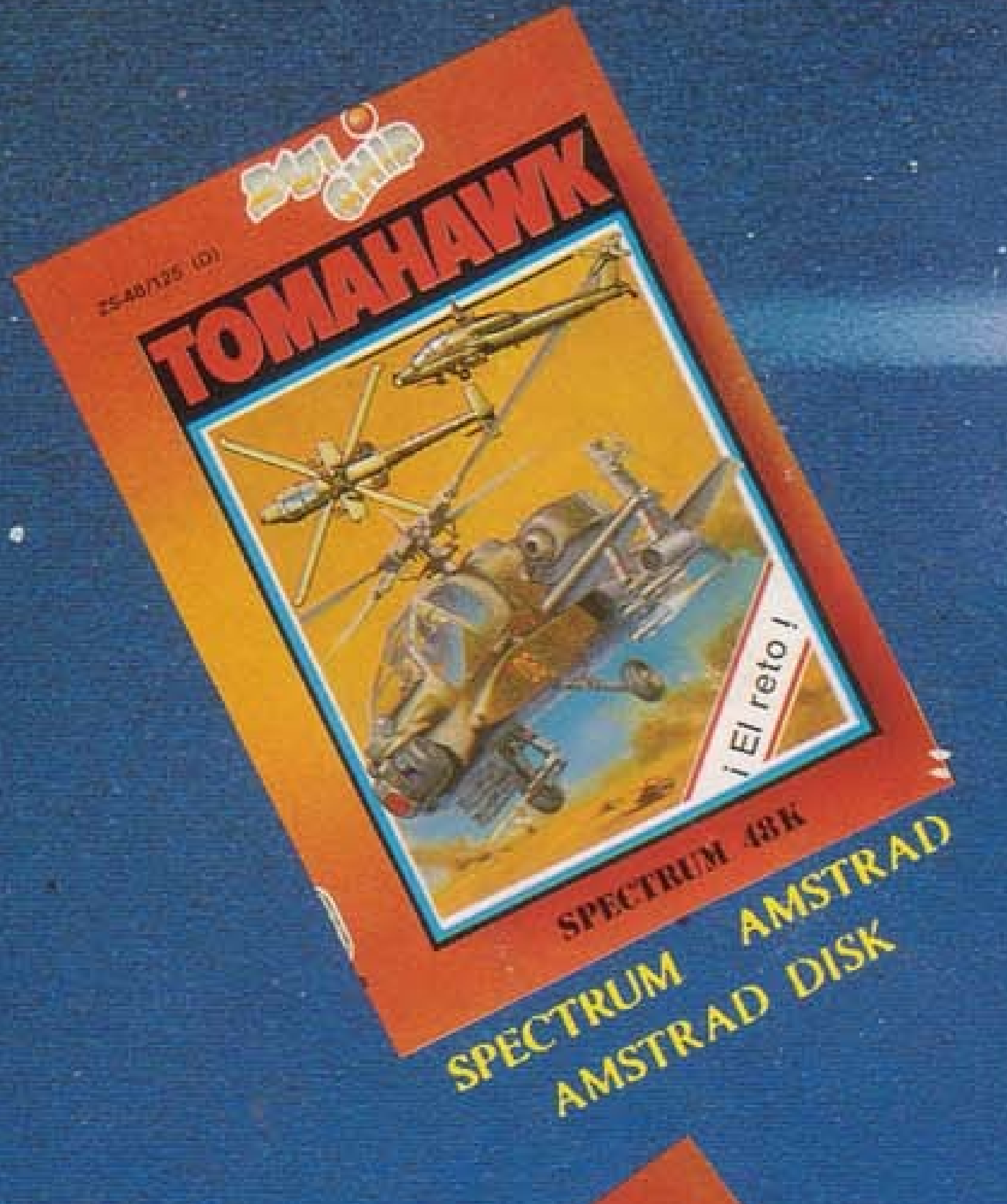
Imagina el planeta Tierra sumido en la radioactividad. Imagina hordas de mutantes luchando a muerte por conseguir alimentos.
Imagina las bandas de asesinos recorriendo ciudades dormidas, multitud de conversores gamma acechando en las esquinas. Un mundo donde vida y muerte se suceden como una danza macabra.



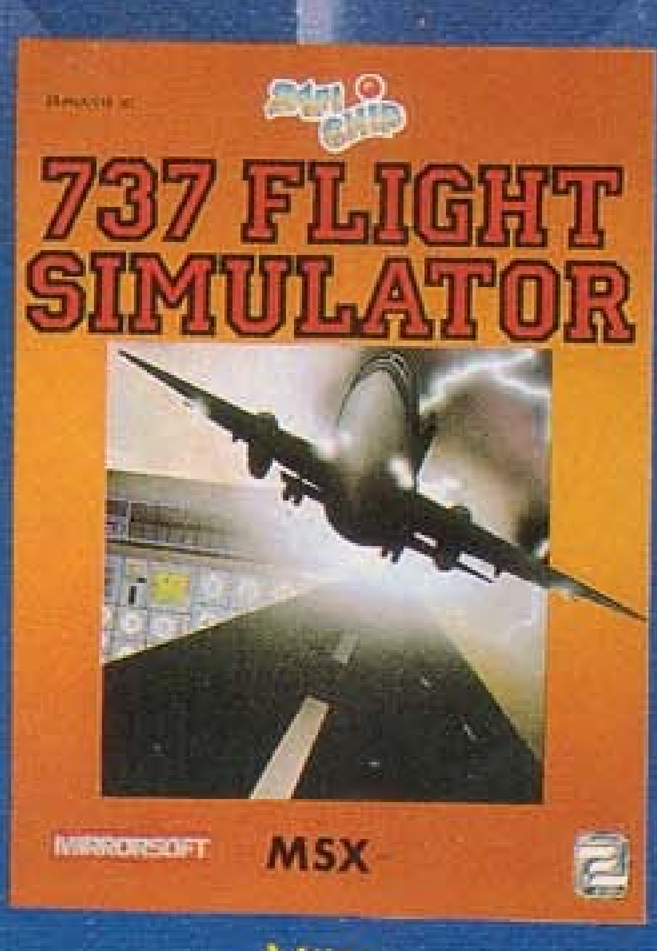
DINAMIC SOFTWARE. P. DE ESPAÑA, 18. TORRE DE MADRID 29-1 · 28008 MADRID · TELEX: 47008
PEDIDOS CONTRAREEMBOLSO: (91) 248 78 87 · DISTRIBUIDORES: (91) 447 34 10

VUELA ALTO

LOS MEJORES SIMULADORES DE VUELO PARA TU ORDENADOR



SPECTRUM AMSTRAD
AMSTRAD DISK



MSX



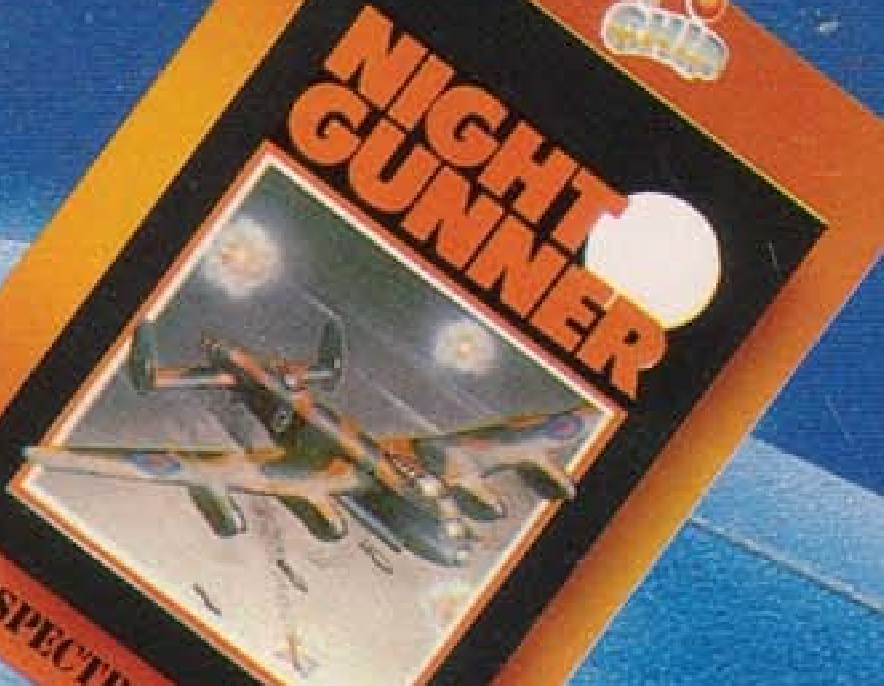
SPECTRUM
COMMODORE

FIGHTER PILOT

SPECTRUM 48K



SPECTRUM COMMODORE
AMSTRAD AMSTRAD DISK



SPECTRUM AMSTRAD
AMSTRAD DISK



ZAFIRO SOFTWARE DIVISION
Paseo de la Castellana, 141 28046 Madrid
Tel. 459 30 04 Telex 22690 ZAFIR E